JDF Specification for Open Source Digital Printing

(Proposed)
Version 1.3

Claudia Alimpich
IBM Printing Systems Division
Boulder CO
303-924-4418
alimpich@us.ibm.com
10/11/2002

1. Changes made to this specification

- **7/7/02**
  1) Initial version of JDF Specification for Open Source Digital Printing.

- **7/26/02 Version 1.1**
  1) Changed Destination or Physical Printer Requested function to have a scope of Page.
  2) Added description to Insert Sheet function that states that insert sheets are included when finishing is specified for related page range.
  3) Added example to Jog Offset function.
  4) Added osdp:PercentRecycled attribute to Media function.
  5) Changed Notification function to allow one or more osdp:Locator attributes.
  6) Increased Number of Copies from 32640 to 32767.
  7) Added Project ID function.
  8) Added Range of Pages to Process function.
  9) Added Angle attribute to Stapling and Stitching function.
  10) Added description to Start, Separator/Slip, and End Sheets function that states that separator sheets are not included when finishing is specified for any part of the job.
  11) Changed Trimming function to have a scope of Page.

- **8/2/02 Version 1.2**
  1) Added Manufacturer attribute to Destination or Physical Printer Requested function.
  2) Changed Fit Policy to have scope of Page.
  3) Removed osdp:HostName attribute from Job Modified By function.
  4) Changed Media Dimension value to be a Double instead of an Integer.
  5) Changed Range of Pages to Process to have scope of Document.
  6) Added JDF to PPML Mapping table to JDF Mapping Addendum.

- **10/11/02 Version 1.3**
  1) Removed most values from osdp:Type attribute for File Type function since they are not generic.
  2) Capitalized values of odsp:Type attribute for File Type function.
  3) Added ProjectID to JDF node in "Structure of an Open Source Digital Printing JDF Job Ticket" section.
  4) Added standalone attribute to XML declaration.
  5) Added LayoutPreparationParams Run partition to Covers function Syntax.
  6) Changed type of ShiftBack attribute in Image Shift Back Side function and ShiftFront attribute in Image Shift Back Side function from Integer to Double.
  7) Moved Name attribute from InsertSheet element in Insert Folded Sheet function to Sheet element that is beneath InsertSheet.
  8) Moved LayoutElementRef beneath partitioned (Run) RunList in Insert Sheet function. Changed Scope from Page to Document.
  9) Changed odsp:Locator attribute in Notification function to allow multiple addresses instead of one. Changed osdp:Locator to be optional with certain values of osdp:ChannelType.
  10) Changed Number of Copies function so that Amount attribute is in ComponentLink instead of Component.
  11) Added note to ProjectID function that in the future there will be new attributes added to AuditPool Created.
  12) Added note to Start First Page stating that it only applies to two sided printing.
  13) Added Media to Start, Separator/Slip, End Sheets function.
  14) Added statement to Start, Separator/Slip, End Sheets function to allow SheetUsage=Slip with SheetType=JobSheet for separator sheets between job copies.

2. Items to add to specification in the future

1) Once it is available, run JDF 1.1 version of CheckJDF against all JDF examples to get rid of syntax errors.

2) Add IPP job submission function for the Linux print system. Add JDF to Linux (IPP) mapping table.

3. Terminology and Definitions

1) A **document** is a file that is included in a **job**. A job consists of one or more documents or files. Some of the JDF defined in this specification can be applied to specific pages in a document, some JDF can be applied to specific documents in a job, and some JDF can only be applied to the entire job. If the JDF can be applied to specific pages in a document, it can also be applied to the entire document, and if the JDF can be applied to specific documents in a job, it can also be applied to the job which means that it will be applied to all of the documents in the job. Even if it is stated that the JDF can be applied to specific pages or documents in the job, the system-specific implementation of JDF makes the final determination of whether the function can be specified for pages, for each document, or the entire job. If the JDF is specified for the document and also for specific pages, the JDF specified for the pages will override the document specification. If the JDF is specified for the entire job and also for a specific document, the JDF specified for the document will override the job specification.

   There are printable and non-printable **documents**. A printable document is a file that contains data that is to be printed and a non-printable document is a file that contains information that is to be used to print the data, like fonts fro example. The function described in the JDF Supported by Open Source Digital Printing section on page 22 only applies to printable documents, except for Document File Name on page 31 and File Type on page 33 which are used to identify a non-printable document for use by the system.

2) A **JDF job ticket** is comprised of one or more **JDF nodes**. The Open Source Digital Printing implementation of a JDF job ticket contains only one JDF node. A JDF node is an XML element and contains **attributes** and **elements**. Elements contain attributes and **sub-elements**. Sub-elements are XML elements that are contained in another element.

3) The Open Source Digital Printing implementation of a JDF job ticket contains a single JDF node that is a combined process node. A **process** is a unit of work or a step in the job workflow. Stitiching, Folding, Trimming, LayoutPreparation, are all examples of processes. A job ticket that contains a **combined process** JDF node simply contains multiple processes.

4) **Resources** are consumed (input) or produced (output) by processes.

5) The term "system" or "system-defined" is used when referring to an operating system-specific implementation of Open Source JDF for Digital Printing.

6) A **sheet** of paper contains one or more finished **pages**. A sheet of paper with no folds consists of 2 finished **pages**. A brochure with one fold inside has 4 finished **pages**.

4. Introduction to JDF

What is JDF?

JDF (Job Definition Format) is an extensible, XML-based job ticket format that is used by the printing industry. It has the ability to unify the pre-press, press, and post-press aspects of a printing job. It provides the means to bridge the communication gap between MIS (Management Information Services) and Production Services. MIS oversees the relationship between all of the units in a workflow and is responsible for planning and controlling the job. Production Services is responsible for the operation of the job. So JDF provides a way to describe a job from the time that a customer places the order until the time that the finished product is placed in the customer's hands.

JDF was developed and is maintained by CIP4 (International Cooperation for Integration of Processes in Prepress, Press, and Postpress) which is an international world wide operating standards body whose purpose is to encourage computer based integration of all processes that have to be considered in the graphics arts industry.

JDF is a vendor-independent XML-based data format. It is not a product or application. It is easily extensible so that it can be tailored to meet vendor-specific requirements and product capabilities. JDF does not dictate that a workflow must be constructed in any pre-specified way. Its flexibility allows JDF to model a custom solution. JDF is equally as effective with a simple system, such as submitting a job to a printer.

Elements in a JDF Job Ticket

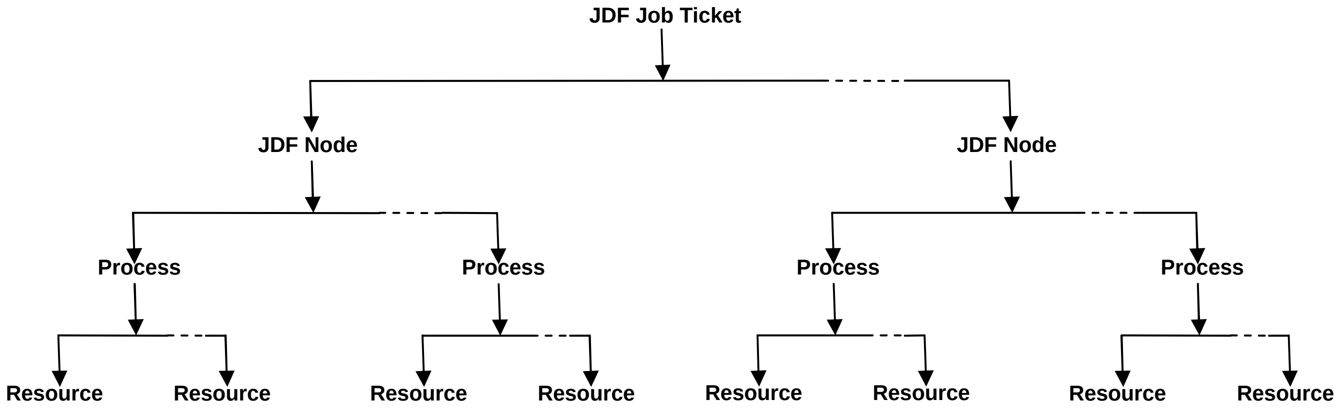
A JDF job ticket is an XML document so it requires that the version of XML and the character set encoding be specified as follows:

```
<?xml version=”1.0” encoding="UTF-8”?>
```

encoding="UTF-8" is the default which specifies compressed Unicode and includes ASCII as a subset.

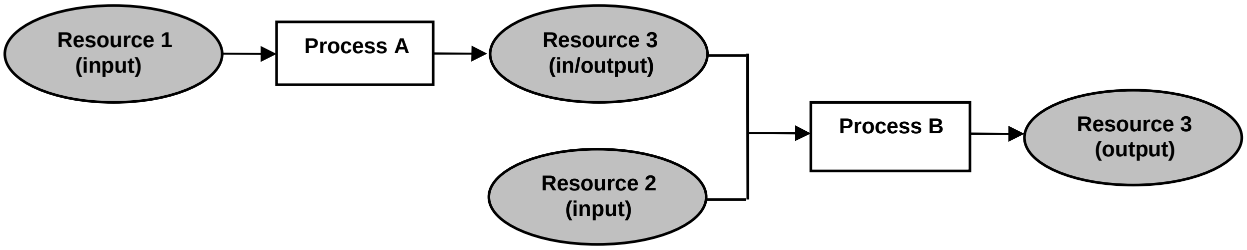
There is a conceptual view of a JDF job ticket and an actual view. The conceptual view is the model on which a JDF job ticket is built where a JDF job ticket is comprised of one or more JDF nodes, each JDF node is comprised of one or more processes, and resources are inputs to and outputs from processes. The actual or physical view of a JDF job ticket is how the document that contains the JDF XML is physically structured and what it contains.

Following is a diagram of the conceptual view of a JDF job ticket:



A JDF node is an XML element that represents an aspect of a job in terms of a process, such as stitching, folding, or trimming; in terms of a product that contributes to the end result, such as a brochure; or in terms of the combination of the two. A process is a unit of work or a step in the job workflow. A JDF node that is comprised of multiple processes is called a combined process JDF node.

Resources are inputs to and outputs from processes that provide details or information to the process or from one process to another process. For example, the StitchingParams resource is an input to the Stitching process and specifies the number of stitches and the placement of the stitches. The Component resource is an output of the DigitalPrinting process and specifies how many copies of the job to produce. The order that the processes are specified in a combined process JDF node is pertinent because it defines the order that the processes are to be executed. So if the job is to be stitched before it is folded as in the case of a saddle stitched booklet, then the Stitching process must be specified before the Folding process.



A JDF node is specified as follows:

<JDF ID="*node ID*" Type="*pre-defined JDF type or any JDF process*"
Types="*list of JDF processes if combined process JDF node*" Status="Ready" Activation="Active"
xmlns="http://www.cip4.org/JDFSchema_1_1" xmlns*:privatens="private XML namespace*"
Version="1.1">

JDF extensibility is implemented using XML namespaces. The URI for the official JDF XML namespace is http://www.cip4.org/JDFSchema_1_1. Private namespaces can be declared and used to extend existing JDF resources by adding private attributes and elements. The namespaces that are to be used by the job ticket are declared in the JDF node and appear as xmlns:osdp="*open source private namespace*" for the Open Source Digital Printing extensions. Namespaces that contain vendor specific attributes and/or values can also be declared.

The version of JDF that is used to generate the job ticket is specified in the outside JDF node by the Version attribute. The version of JDF that is supported by Open Source Digital Printing is 1.1.

A JDF node contains attributes and elements. Elements contain attributes and sub-elements. Sub-elements are XML elements that are contained in another element. A JDF node can contain any or all of the following elements:

- **Comment** - Human readable comment or description of the job ticket. A JDF node can contain 0 or more Comment elements.
- **AncestorPool** - Contains 1 or more Ancestor elements and is only required for a spawned job. Spawning is the process of extracting a JDF sub-node from a job and creating a new, complete JDF document. An Ancestor element reflects the family tree of the spawned job. A JDF node can contain 0 or 1 AncestorPool elements.
- **CustomerInfo** - Information about the customer who ordered the job. A JDF node can contain 0 or 1 CustomerInfo elements.
- **NodeInfo** - Information about planned scheduling and message routing. A JDF node can contain 0 or 1 NodeInfo elements.
- **ResourceLinkPool** - Links the JDF node to the Resources it uses. Also identifies the processes each Resource is an input to or output from and whether the Resource is an input or output. A JDF node can contain 0 or 1 ResourceLinkPool elements.
- **ResourcePool** - Contains 0 or more Resource elements where a Resource element is something that is consumed or produced. There are many possible Resources defined by JDF, many of which will not be supported by a specific implementation of JDF. A JDF node can contain 0 or 1 ResourcePool elements.

- **AuditPool** - Contains post-facto recorded results. A JDF node can contain 0 or 1 Audit Pool elements.
- **StatusPool** - Describes the status of a JDF node that processes partitioned Resources. A partitioned Resource is a Resource that has individual nested parts. A JDF node can contain 0 or 1 StatusPool elements.
- **JDF** - A nested JDF node. A JDF node can contain 0 or more JDF nodes.

Structure of a JDF Job Ticket

The actual view of a JDF job ticket is the physical structure or content of the job ticket. Following is a sample JDF job ticket document that contains all of the elements described in the previous section:

```
<?xml version=”1.0” encoding=”UTF-8”?>
<JDF ID=”outer node ID” Type=“pre-defined JDF type or any JDF process“
        Types=”list of JDF processes if combined process JDF node” Status=”Ready” Activation=”Active”
        xmlns="http://www.cip4.org/JDFSchema_1_1" xmlns:privatens="private XML namespace"
        Version=”1.1”>
        <Comment …>
        </Comment>
        <AncestorPool >
                <Ancestor …/>
                 <Ancestor …/>
        </AncestorPool>
        <CustomerInfo …/>
        <NodeInfo …/>
        <ResourceLinkPool>
                <Link for 1st Resource and processes that use it appear here />
                <Link for 2nd Resource and processes that use it appear here />
        </ResourceLinkPool>
        <ResourcePool>
                <1st Resource appears here />
                <2nd Resource appears here />
        </ResourcePool>
        <AuditPool>
                <An Audit element appears here />
        </AuditPool>
        <StatusPool>
                <Status=”Ready”/>
        </StatusPool>
        <JDF ID=”inner node ID” Type=”pre-defined JDF type or any JDF process“
                Types=”list of JDF processes in combined process JDF node” Status=”Ready”>
                Elements contained in this nested JDF node appear here. Any of the elements shown
                above in the outside JDF node can appear here.
        </JDF>
</JDF>
```

Comments in a Job Ticket

There are two types of comments that can appear in a JDF job ticket:

1. The following notation allows comments to be added inside of the job ticket. These comments aren’t available outside of the job ticket. They are just comments to assist anyone reading the JDF in the job ticket.

    ```
    <!-- The following specifies how to print duplex. -->
    ```

2. The following is a comment that can be extracted from the JDF job ticket and interpreted by the consumer of the job ticket:

```
<Comment Name=”Description” A description of this job ticket. </Comment>
```

5. Use of JDF by Open Source Digital Printing

How Open Source Digital Printing will use JDF

JDF was designed to describe a job from conception through delivery so the job ticket can be used for the entire workflow. JDF was also designed so that it is valid to use only the JDF that defines part of the workflow. Initially, Open Source Digital Printing will use JDF to define the function or attributes necessary to describe what a printer device and the software that processes the data to get it ready to print can do, like printing, hole making, folding, imposition, etc. The Open Source Digital Printing implementation of JDF will use a combined process JDF node, so the outside JDF node will have a Type of Combined and the nested JDF node as it appears in the above sample is not necessary.

JDF is a very flexible and extensible data format, so sometimes there may be a number of ways to define or represent the same function in JDF. This specification will define the Open Source Digital Printing preferred method and syntax to define the JDF that is interpreted by Open Source Digital Printing software that use a JDF job ticket, so the remainder of this specification will only define the JDF and the JDF syntax that is supported by Open Source Digital Printing. The CIP4 JDF specification that defines all valid JDF and JDF syntax can be found at www.cip4.org.

The JDF that is described in this document is the JDF that is supported by Open Source Digital Printing, but not all of the JDF described is supported by all of the Open Source Digital Printing software products that support the JDF job ticket. It is up to the individual implementations (referred to as a "system" in this specification) of the JDF job ticket to determine which JDF is supported. Some Open Source Digital Printing products may not support the function as defined by a specific JDF construct. The same can be said for the allowed values listed for a JDF attribute. The allowed values listed are all of the values that are valid for the Open Source Digital Printing JDF job ticket. Not all Open Source Digital Printing products that support the JDF job ticket support all of the allowed values listed. It is up to the individual implementations of the JDF job ticket to determine if a specific value is valid.

Elements of an Open Source Digital Printing JDF Job Ticket

Some of the elements that a JDF node can contain as shown in the above sample are not necessary in order to define the printing function as supported by Open Source Digital Printing.

The JDF node in a JDF job ticket supported by Open Source Digital Printing contains one of each of the following elements, except there can be zero or more Comment elements and Comment, CustomerInfo, NodeInfo, and AuditPool are optional. They are optional because they do not have to exist in the JDF job ticket if the function that they contain is not specified for the job.

- **Comment**
- **CustomerInfo**
- **NodeInfo**
- **ResourceLinkPool**
- **ResourcePool**
- **AuditPool**

Structure of an Open Source Digital Printing JDF Job Ticket

An Open Source Digital Printing JDF job ticket can contain all of the elements described in the previous section. The structure and content of an Open Source Digital Printing job ticket is shown as follows. The specific order of the elements belonging to a JDF node, the resources listed in the ResourcePool, and the resource links in the

ResourceLinkPool is not required by JDF. Only specific processes and resources are supported by the Open Source Digital Printing JDF, all of which are listed below.

```
<?xml version=”1.0” encoding=”UTF-8” standalone="yes" ?>
<JDF ID=”J1” Type=”Combined” Types=”LayoutPreparation Imposition Interpreting Rendering Screening
    DigitalPrinting Stitching Folding Trimming HoleMaking Cutting Perforating” Status=”Ready”
    Activation=”Active” ProjectID="text describing the application submitting the job"
    xmlns="http://www.cip4.org/JDFSchema_1_1"
    xmlns:osdp="private Open Source Digital Printing XML namespace"  Version=”1.1”>
    <Comment ...>
    </Comment>
    <CustomerInfo … />
    <NodeInfo ... />
    <ResourceLinkPool>
        <RunListLink rRef="ID of RunList resource" ...
           Usage=”Input” CombinedProcessIndex=”0 1” />
        <LayoutPreparationParamsLink rRef="ID of LayoutPreparationParams resource"   …
           Usage=”Input” CombinedProcessIndex=”0” />
        <OSDPInterpretingParamsLink rRef="ID of OSDPInterpretingParams resource" …
           Usage=”Input” CombinedProcessIndex=”2” />
        <ColorantControlLink rRef="ID of ColorantControl resource" …
           Usage=”Input” CombinedProcessIndex=”2 5” />
        <RenderingParamsLink rRef="ID of RenderingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”3” />
        <ScreeningParamsLink rRef="ID of ScreeningParams resource"  …
           Usage=”Input” CombinedProcessIndex=”4” />
        <ComponentLink rRef="ID of Component resource"  ...
           Usage=”Output” CombinedProcessIndex=”5” />
        <DeviceLink rRef="ID of Device resource"  ...
           Usage=”Input” CombinedProcessIndex=”5” />
        <MediaLink rRef="ID of Media resource"  ...
           Usage=”Input” CombinedProcessIndex=”5” />
        <DigitalPrintingParamsLink rRef="ID ofDigitalPrintingParms resource"  ...
           Usage=”Input” CombinedProcessIndex=”5”/>
        <BindingIntentLink rRef="ID of BindingIntent resource"  ...
           Usage=”Input” CombinedProcessIndex=”5”/>
        <StitchingParamsLink rRef="ID of StitchingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”6” />
        <FoldingParamsLink rRef="ID of FoldingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”7” />
        <TrimmingParamsLink rRef="ID of TrimmingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”8” />
        <HoleMakingParamsLink rRef="ID of HoleMakingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”9” />
        <CuttingParamsLink rRef="ID of CuttingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”10” />
        <PerforatingParamsLink rRef="ID of PerforatingParams resource"  …
           Usage=”Input” CombinedProcessIndex=”11” />
    </ResourceLinkPool>
    <ResourcePool>
        <LayoutElement … />
        <RunList ... >
            <LayoutElementRef ... />
        </RunList>
        <LayoutPreparationParams ... >
            <MediaRef ... />
        </LayoutPreparationParams>
        <OSDPInterpretingParams … />
        <ColorantControl …/>
        <RenderingParams …/>
        <ScreeningParams … />
        <Component ... />
        <Device ... />
        <Media ... />
        <DigitalPrintingParams ... >
            <MediaRef ... />
        </DigitalPrintingParams>
        <BindingIntent … />
        <StitchingParams … />
        <FoldingParams … />
        <TrimmingParams … />
        <HoleMakingParams … />
        <CuttingParams … />
        <PerforatingParams … />
    </ResourcePool>
    <AuditPool>
        <Created … />
    </AuditPool>
</JDF>
```

The processes listed above as the values for the JDF node's Types attribute are the processes that will be executed. If a process is not to be executed, the process does not appear as a value for Types and no resources are defined in the ResourcePool or ResourceLinkPool for that process. For example, if the job is not to be stitched, the Stitching process does not appear as a value for Types, there is no StitchingParams in the ResourcePool or StitchingParamsLink in the ResourceLinkPool.

The order of the processes listed in the JDF node's Types attribute is the order that they will be executed, so it is the order that the work will be performed (for example, Stitching will occur before Folding as shown above). In some cases the output resource of one process is the input resource to another process, so the order of the processes is determined by this. The LayoutPreparation, Imposition, Interpreting, Rendering, and DigitalPrinting processes will always be present in an Open Source Digital Printing job ticket and in the order as shown above. The Screening process is optional and if present will appear after the Rendering process and before the DigitalPrinting process. All other processes are optional and their order can be changed to obtain the desired result. It is important to note that when a resource is an output from one resource and an input to another, the resource does not appear in the ResourcePool or ResourceLinkPool.

A resource and its respective resource link must be defined in the ResourcePool and ResourceLinkPool if the processes that it is an input to or output from are listed in the JDF node's Types attribute and the processes require that it be defined. Following is a table that shows the resources that are required by or optional for each of the processes:

Process	**Required Input Resources**	**Optional Input Resources**	**Required Output Resources**
LayoutPreparation	RunList	LayoutPreparationParams	---
Imposition	RunList	---	---
Interpreting	---	OSDPInterpretingParams ColorantControl	---
Rendering	---	RenderingParams	---
Screening	---	ScreeningParams	---

DigitalPrinting	---	DigitalPrintingParams ColorantControl Device Media BindingIntent	Component
Stitching	StitchingParams	---	---
Folding	FoldingParams	---	---
Trimming	TrimmingParams	---	---
HoleMaking	HoleMakingParams	---	---
Cutting	CuttingParams	---	---
Perforating	PerforatingParams	---	---

As mentioned previously, the LayoutPreparation, Imposition, Interpreting, Rendering, and DigitalPrinting processes will always be present in the Open Source Digital Printing job ticket. So, the RunList and Component resources and their respective resource links are always present as well.

There are resources defined in the ResourcePool that are not inputs to or outputs from processes, but are merely used by another resource that is in the ResourcePool. The LayoutElement is an example of such a resource. It is defined in the ResourcePool and referenced by the RunList resource, but is not an input to or output from a process. Note that because LayoutElement is not an input to or output from a process, it does not have an associated resource link in the ResourceLinkPool.

Each of the ResourceLink elements in the ResourceLinkPool shown above have a CombinedProcessIndex attribute. The CombinedProcessIndex attribute is an integer that is the zero-based index into the JDF node's Types attribute. It identifies the processes that the resource is an input to or an output from. For example, the CombinedProcessIndex for RunListLink has a value of "0 1", so the RunList resource is an input to both the LayoutPreparation and Imposition processes.

6. Fundamentals of JDF

JDF Node

One JDF node appears at the top of the job ticket and for the Open Source Digital Printing implementation always has a Type of "Combined" since the combined process model will be used to described the Open Source Digital Printing job ticket. For the Open Source Digital Printing implementation the Status of the JDF node is always "Ready". The value of Activation refers to the held status of the job (see Hold Job on page 36).The official JDF XML namespace and the private Open Source Digital Printing namespace are also declared in the JDF node. The Open Source Digital Printing namespace is identified by "xmlns:osdp".

Resources and ResourcePool

Resources are elements that are consumed (input) or produced (output) by processes The processes in a combined process JDF node use resources to produce other resources. For example, the Link for the Component resource specifies the number of copies of a job to print, so it is produced by the DigitalPrinting process. The DigitalPrintingParams resource specifies what output bin to use, so it is consumed by the DigitalPrinting process.

Each Resource declaration must contain the following attributes:

- **ID** -A Unique identifier of the resource. An ID cannot contain spaces or blanks. It can contain letters, numbers, underscores (_), dashes (-), periods (.), and colons (:). The first character of the ID must be a letter, underscore, or colon.
- **Class** -The Open Source Digital Printing implementation of JDF requires that one of the following abstract resource types be specified and is dependent on the resource being declared:
    1) Implementation

2) Parameter
3) Quantity
4) Consumable

- **Status -** Indicates whether the resource is available for use. For the Open Source Digital Printing implementation of JDF the Status is always “Available”.

All Resources are defined and reside in the ResourcePool.

Resource Links and ResourceLinkPool

Each JDF node contains a ResourceLinkPool element that in turn contains a resource link element for each resource that is used by the processes in the JDF node. The resource link elements also define whether the resources are inputs (consumed) or outputs (produced) and what processes they are inputs to or outputs from. The rRef attribute of each resource link element references a Resource that is defined in the ResourcePool.

In the following example, the ComponentLink element is located in the ResourceLinkPool and uses the rRef attribute to reference the Component resource that is defined in the ResourcePool and has an ID of “Amount”.

```
...
<ResourceLinkPool>
        <ComponentLink rRef=”Amount” Usage=”Output” CombinedProcessIndex=”5” Amount="5" />
        ...
</ResourceLinkPool>
<ResourcePool>
        <Component ID=”Amount” Class=”Quantity” Status=”Available”
                ComponentType=”FinalProduct” />
                ~~Amount=”5” />~~
...
</ResourcePool>
...
```

Only those Resources that appear in the ResourceLinkPool are used when the job is processed. So if there are Resources defined in the ResourcePool that should be inputs to or outputs from the processes specified in the Types attribute for the JDF node, these Resources must each have a ResourceLink in the ResourceLinkPool or they will not be used.

Following is the list of all possible ResourceLinks that can exist in the ResourceLinkPool supported by Open Source Digital Printing. It is possible that some of the processes supported by Open Source Digital Printing may not be included in the job ticket because the function is not necessary or are not supported by the Open Source Digital Printing product, so the resources required by that process will not be defined in the ResourcePool, and the resource link for that resource will not appear in the ResourceLinkPool. For example, if the job is not to be stitched, the Stitching process is not needed. “Stitching” will not appear in the Types attribute of the JDF node, there will not be a StitchingParams resource, and a StitchingParamsLink is not needed.

<RunListLink rRef=”*ID of RunList resource being referenced*” Usage=”Input”
CombinedProcessIndex=”0 1” />
<LayoutPreparationParamsLink rRef=”*ID of LayoutPrepartionParams resource being referenced*” Usage=”Input”
CombinedProcessIndex=”0” />
<OSDPInterpretingParamsLink rRef=”*ID of InterpretingParams resource being referenced*” Usage=”Input”
CombinedProcessIndex=”2” />
<RenderingParamsLink rRef=”*ID of RenderingParams resource being referenced*” Usage=”Input”
CombinedProcessIndex=”3” />
<ScreeningParamsLink rRef=”*ID of ScreeningParams resource being referenced*” Usage=”Input”
CombinedProcessIndex=”4” />
<ComponentLink rRef=”*ID of Component resource being referenced*” Usage=”Output”
CombinedProcessIndex=”5” />

```
<DeviceLink rRef=”ID of Device resource being referenced” Usage=”Input”
        CombinedProcessIndex=”5” />
<MediaLink rRef=”ID of Media resource being referenced” Usage=”Input”
        CombinedProcessIndex=”5” />
<DigitalPrintingParamsLink rRef=”ID of DigitalPrintingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”5” />
<BindingIntentLink rRef=”ID of DigitalPrintingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”5” />
<StitchingParamsLink rRef=”ID of StitchingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”6” />
<FoldingParamsLink rRef=”ID of FoldingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”7” />
<TrimmingParamsLink rRef=”ID of TrimmingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”8” />
<HoleMakingParamsLink rRef=”ID of HoleMakingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”9” />
<CuttingParamsLink rRef=”ID of Cutting resource being referenced” Usage=”Input”
        CombinedProcessIndex=”10” />
<PerforatingParamsLink rRef=”ID of PerforatingParams resource being referenced” Usage=”Input”
        CombinedProcessIndex=”11” />
```

In the Open Source Digital Printing implementation of the JDF job ticket there can be zero or one of each of the above types of resource links in the ResourceLinkPool, except for the MediaLink. Multiple Media resources can be inputs to the DigitalPrinting process, so there can be multiple Media resources defined in the ResourcePool, and multiple MediaLinks in the ResourceLinkPool.

Resource References

In some cases, it is possible that multiple resources reference (use or contain) another resource. Instead of defining the referenced resource twice and risking that when updating one definition, the other definition is forgotten and not updated, a resource reference (Inter-Resource Linking) can be used. Using a resource reference is also a way to isolate a resource definition from the resources that use or contain it.

The Open Source Digital Printing implementation of JDF uses resource references instead of imbedding resources inside of other resources.

In the following example the Media resource is declared in the ResourcePool and the DigitalPrintingParams and LayoutPreparationParams resources each use a reference to refer to it. Note the addition of the “Ref” suffix when the DigitalPrintingParams and LayoutPreparationParams resources reference the Media resource and the use of the rRef attribute to identify the Media resource by its ID attribute. It is clear that the same Media resource is being referenced by both DigitalPrintingParams and LayoutPreparationParams, and any updates to the Media resource will only have to be made in one place.

```
.....
<ResourcePool>
        ...
        <Media ID=”TopTray” Class=”Parameter” Status=”Available” >
                 <Location LocationName=”top”/>
        </Media>
        <LayoutPreparationParams ID=”LPP” Class=”Parameter” Status=”Available” >
                <MediaRef rRef=”TopTray” />
        </LayoutPreparationParams>
        <DigitalPrintingParams ID=”DPP” Class=”Parameter” Status=”Available” >
                <MediaRef rRef=”TopTray” />
        </DigitalPrintingParams>
        ...
</ResourcePool>
```

...

Partitionable Resources

Part of a resource may have to be referenced separately from another part of the resource. For example, a job may consist of multiple documents or files. A partitionable resource contains nested resource elements, each with the same name as the resource.

The Open Source Digital Printing implementation of JDF uses the partitioning of resources to accomplish the following:

When the JDF is to be different for individual documents (files) in the job. In this case **PartIDKeys="Run"** is specified in the parent resource and **Run="*unique document identifier*"** is specified in each child.

When the JDF is to be different for specific pages of a document (file). In this case the **PartIDKeys="RunIndex"** is specified in the parent resource and **RunIndex="*list of individual pages and/or page ranges*"** is specified in each child.

Multiple values for PartIDKeys can be specified if nested partitioning is necessary. For example, when the JDF is to be different for specific pages of a specific document (file) in a job. In this case the **PartIDKeys="Run RunIndex"** is specified in the parent resource. **Run="*unique document identifier*"** is specified in each child and then **RunIndex="*list of individual pages and/or page ranges*"** is specified in a nested child.

An example of a) would be if a job contains multiple documents that are to be printed, where the first document is to be printed on one side of the sheet and the second document on both sides of the sheet. The RunList and the LayoutPreparationParams resources are the resources that are partitioned. Note that the parent RunList and LayoutPreparationParams includes PartIDKeys="Run" which specifies that partitioning is being done based on the Run attribute and that each child RunList and LayoutPreparationParams includes a unique value for its Run attribute that references a specific document. So, the Run attribute is used to distinguish between the documents in a job.

```
...
<ResourcePool>
        ...
        <RunList ID="RunList" Class="Parameter" Status="Available" PartIDKeys="Run" >
                <RunList Run="1" >
                        <LayoutElement ID="InputFile1" Class="Parameter" Status="Available" >
                                <FileSpec URL="file:///d:/printFile1.afp" />
                        </LayoutElement>
                </RunList>
                <RunList Run="2" >
                        <LayoutElement ID="InputFile2" Class="Parameter" Status="Available" >
                                <FileSpec URL="file:///d:/printFile2.afp" />
                        </LayoutElement>
                </RunList>
        </RunList>
        <LayoutPreparationParams ID="LPP" Class="Parameter" Status="Available" PartIDKeys="Run" >
                <LayoutPreparationParams Run="1" Sides="OneSidedFront" />
                <LayoutPreparationParams Run="2" Sides="TwoSidedFlipY" />
        </LayoutPreparationParams>
        …
</ResourcePool>
…
```

An example of b) would be if specific pages and page ranges are to be printed on the front side of the sheet and the rest of the pages are to be printed on both sides of the sheet. The LayoutPreparationParams resource is the resource that is partitioned. Note that the parent LayoutPreparationParms includes PartIDKeys="RunIndex" which specifies that partitioning is being done based on the RunIndex attribute and that each child LayoutPreparationParams includes page numbers and page ranges as values for its RunIndex attribute. Note that the pages referenced by RunIndex are the summation of the page numbers in both documents, so the page numbers of the second file starts where the page numbers of the first file ends.

```
    …
    <ResourcePool>
        …
        <RunList ID=”RunList” Class=”Parameter” Status=”Available” PartIDKeys=”Run” >
            <RunList Run=”1” >
                <LayoutElement ID=”InputFile1” Class=”Parameter” Status=”Available” >
                    <FileSpec URL=”file:///d:/printFile1.afp” />
                </LayoutElement>
            </RunList>
            <RunList Run=”2” >
                <LayoutElement ID=”InputFile2” Class=”Parameter” Status=”Available” >
                    <FileSpec URL=”file:///d:/printFile2.afp” />
                </LayoutElement>
            </RunList>
        </RunList>
        <LayoutPreparationParams ID=”LPP” Class="Parameter" Status="Available"
            PartIDKeys="RunIndex" >
            <LayoutPreparationParams RunIndex="5 6 8~12 20~-1" Sides="OneSidedFront" />
            <LayoutPreparationParams RunIndex="0~4 7 13~19" Sides=”TwoSidedFlipY”  />
        </LayoutPreparationParams>
        …
    </ResourcePool>
    …
```

An example of c) would be if specific pages and page ranges to be printed need to be referenced with respect to the beginning of each document in the job and not as a summation of the pages in the two documents. In this case the LayoutPreparationParams that is partitioned using RunIndex is nested in the LayoutPreparationParams that is partitioned using Run. The Run partition refers to the specific document and then the nested RunIndex partition refers to the pages in that document.

```
    …
    <ResourcePool>
        …
        <RunList ID=”RunList” Class=”Parameter” Status=”Available” PartIDKeys=”Run” >
            <RunList Run=”1” >
                <LayoutElement ID=”InputFile1” Class=”Parameter” Status=”Available” >
                    <FileSpec URL=”file:///d:/printFile1.afp” />
                </LayoutElement>
            </RunList>
            <RunList Run=”2” >
                <LayoutElement ID=”InputFile2” Class=”Parameter” Status=”Available” >
                    <FileSpec URL=”file:///d:/printFile2.afp” />
                </LayoutElement>
            </RunList>
        </RunList>
        <LayoutPreparationParams ID=”LPP” Class="Parameter" Status="Available"
            PartIDKeys="Run RunIndex" >
            <LayoutPreparationParams Run="1" />
                <LayoutPreparationParams RunIndex="5 6 8~-1" Sides="OneSidedFront" />
                <LayoutPreparationParams RunIndex="0~4 7 " Sides=”TwoSidedFlipY”  />
            <LayoutPreparationParams Run="2" />
                <LayoutPreparationParams RunIndex="0~6" Sides="OneSidedFront" />
                <LayoutPreparationParams RunIndex="7~-1" Sides=”TwoSidedFlipY”  />
        </LayoutPreparationParams>
        …
```

```
</ResourcePool>
…
```

Using Resource References and Partitionable Resources

Following is an example of using both resource references and partitioned resources.

Resource references are used to refer to the four LayoutElement resources that identify the files that are part of the job. There are four RunList partitions, each of which contains a reference to a LayoutElement that resides in the ResourcePool.

There are three Media resources that are referenced by a DigitalPrintingParams resource. The DigitalPrintingParams resource has been partitioned so that the four input files can be printed on media pulled from different input trays. The first file is printed using the media in the top tray, the second file is printed using the media in the middle tray, the third file is printed using the media in the bottom tray, and the first and last pages of the fourth file are printed using the top tray and the rest of the pages using the middle tray.

```
.....
<ResourcePool>
    ...
    <LayoutElement ID=”InputFile1” Class=”Parameter” Status=”Available” >
        <FileSpec URL=”file:///d:/printFile1.afp” />
    </LayoutElement>
    <LayoutElement ID=”InputFile2” Class=”Parameter” Status=”Available” >
        <FileSpec URL=”file:///d:/printFile2.afp” />
    </LayoutElement>
    <LayoutElement ID=”InputFile3” Class=”Parameter” Status=”Available” >
        <FileSpec URL=”file:///d:/printFile3.afp” />
    </LayoutElement>
    <LayoutElement ID=”InputFile4” Class=”Parameter” Status=”Available” >
        <FileSpec URL=”file:///d:/printFile4.afp” />
    </LayoutElement>
    <RunList ID=”RunList” Class=”Parameter” Status=”Available” PartIDKeys=”Run RunIndex” >
        <RunList Run=”1” >
            <LayoutElementRef rRef=”InputFile1” />
        </RunList>
        <RunList Run=”2” >
            <LayoutElementRef rRef=”InputFile2” />
        </RunList>
        <RunList Run=”3” >
            <LayoutElementRef rRef=”InputFile3” />
        </RunList>
        <RunList Run=”4” >
            <LayoutElementRef rRef=”InputFile4” />
        </RunList>
    </RunList>
    <Media ID=”TopTray” Class=”Parameter” Status=”Available”>
        <Location LocationName=”top” />
    </Media>
    <Media ID=”MiddleTray” Class=”Parameter” Status=”Available” >
        <Location LocationName=”middle” />
    </Media>
    <Media ID=”BottomTray” Class=”Parameter” Status=”Available” >
        <Location LocationName=”bottom” />
    </Media>
```

```
        <!-- Partition DigitalPrintingParams by both Run and RunIndex so that each file in the job and the
        pages in each file can be individually referenced -->
        <DigitalPrintingParams ID=”DPP” Class=”Parameter” Status=”Available”
            PartIDKeys=”Run RunIndex”>
            <!-- The top tray is the default tray to be used, so it is specified in the parent
            DigitalPrintingParams. It will be used by the 1st file in the job since there is not a partition
            for the 1st file. -->
            <MediaRef rRef=”TopTray” />
            <!-- The middle tray will be used to print the file identified by Run 2, which is the 2nd
            file. -->
            <DigitalPrintingParams Run=”2”>
                <MediaRef rRef=”MiddleTray” />
            </DigitalPrintingParams>
            <!-- The bottom tray will be used to print the file identified by Run 3, which is the 3rd
            file. -->
            <DigitalPrintingParams Run=”3”>
                <MediaRef rRef=”BottomTray” />
            </DigitalPrintingParams>
            <!--The top tray will be used to print the first and last pages of the file identified by Run 4
            and the middle tray will be used to print the rest of the pages in the file. -->
            <DigitalPrintingParams Run=”4”>
                <DigitalPrintingParams RunIndex=”0 -1”
                    <MediaRef rRef=”TopTray” />
                <DigitalPrintingParams RunIndex=”1~-2”
                    <MediaRef rRef=”MiddleTray” />
            </DigitalPrintingParams>
        </DigitalPrintingParams>
    </ResourcePool>
    ...
```

Resource Update

ResourceUpdate is an abstract element that optionally contains any of the attributes and elements valid for the Resource that it resides in. Mandatory attributes and elements of resources are optional in the respective ResourceUpdate. In addition, a ResourceUpdate defined within a Resource must contain a unique *UpdateID*. The UpdateID is referenced from outside of the job ticket. The references to ResourceUpdate elements will update the current state of the resource. A ResourceUpdate is specified by adding the suffix “Update” to the Resource name. There can be one or more ResourceUpdate elements defined for each Resource that is defined in a job ticket.

ResourceUpdate elements can only be referenced from outside of the job ticket. Currently only PPML has the ability to reference ResourceUpdate elements. When a ResourceUpdate is referenced from outside of the job ticket, e.g., from a PPML TICKET_REF element, ONLY those elements that are explicitly specified within the ResourceUpdate will be updated.

In the following example the DigitalPrintingParams resource initially specifies that no collation and jog offset be performed. To perform sheet collation, the outside control will reference the DigitalPrintingParamsUpdate that has the UpdateID of CollateEnabled. To turn collation off again, the outside control will reference the DigitalPrintingParamsUpdate that has the UpdateID of CollateDisabled. The same applies when the outside control turns jog offset on and off. Because PageDelivery resides in the DigitalPrintingParams resource specification, its setting is maintained when the DigitalPrintingParamsUpdate elements are referenced.

```
<DigitalPrintingParams ID="InitialDigitalPrintingParams" Class="Parameter" Status="Available"
    PageDelivery=”SameOrderFaceUp” Collate="None" >
    <!-- Collate choices -->
    <DigitalPrintingParamsUpdate UpdateID="CollateEnabled" Collate="Sheet" />
    <DigitalPrintingParamsUpdate UpdateID="CollateDisabled" Collate="None" />
```

```
		<!-- Jog Offset choices -->
		<DigitalPrintingParamsUpdate UpdateID="JogOffsetEnabled" >
				<Disjointing OffsetDirection="Alternate" />
		</DigitalPrintingParamsUpdate>
		<DigitalPrintingParamsUpdate UpdateID="JogOffsetDisabled" >
				<Disjointing OffsetDirection="None" />
		</DigitalPrintingParamsUpdate>
</DigitalPrintingParams>
```

7. JDF Supported by Open Source Digital Printing

This section describes the function that is supported by the Open Source Digital Printing implementation of JDF. Each function has a short description, **Scope**, **Syntax**, **Type**, and **Allowed Values**.

- **Scope** of the JDF identifies whether the function defined by the JDF applies to the entire job, to specific documents in the job, or to specific pages in a document. JDF that applies to specific pages in a document can also be applied to a specific document or all documents in a job. JDF that applies to specific documents in the job can also be specified so that it applies to all documents in a job. JDF function that has a scope of job cannot be specified to apply to specific pages in a document or to apply to a specific document in the job.

- **Syntax** shows how the JDF for the function must appear.

  For the JDF function that can be applied to specific pages in a document in the job, the **Syntax** section shows how the JDF would appear when the JDF applies to specific pages in a document in the job. The JDF can also be applied to a specific document or all documents in a job that contains multiple documents (files) by making the following changes to the **Syntax**:

  1) Remove “RunIndex” from the "PartIDKeys" attribute in the parent resource specification. The existence of the PartIDKeys attribute with the “RunIndex” value signifies that the resource is to be partitioned by specific pages or page ranges.
  2) Remove the child’s children (partitioned) resource specification.

For example, the following specifies to print pages 0 through 5 on the front side of the sheet and pages 6 through the end of a document on two sides of the sheet:

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run RunIndex" >
    <LayoutPreparationParams Run="1"
        <LayoutPreparationParams RunIndex="0~5"
            Sides="OneSidedFront" />
        </LayoutPreparationParams>
        <LayoutPreparationParams RunIndex="6~-1"
            Sides="TwoSidedFlipY" />
        </LayoutPreparationParams>
</LayoutPreparationParams>
```

After applying 1) and 2) from above, the following specification is for a document in the job:

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run" >
    <LayoutPreparationParams Run="1"
        Sides="OneSidedFront" />
    </LayoutPreparationParams>
</LayoutPreparationParams>
```

To apply the JDF function to the entire job make the following changes to the **Syntax**:

3) Remove the "PartIDKeys" attribute from the parent resource specification. The existence of the PartIDKeys attribute with the “Run” value signifies that the resource is to be partitioned by documents.
4) Remove the child (partitioned) resource specification.

After applying 3) and 4) from above, the following specification is for the entire job:

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available"
```

Sides="OneSidedFront" />

- **Type** indicates whether the function can be specified only at job submission time and not modified after that point (Initially settable), at job submission time and can be modified after the job has been submitted (Resettable), or if it cannot be specified at job submission time or modified afterwards (Non-settable). **Type** also indicates whether the function can be specified with one value (single-valued) or multiple values (multi-valued).

- **Allowed Values** identifies the data type of each JDF attribute and the values that are allowed to be specified for each JDF attribute. The allowed values listed are all of the values that are valid for the Open Source Digital Printing JDF job ticket. Not all Open Source Digital Printing products that support the JDF job ticket support all of the allowed values listed. It is up to the individual system implementations of the JDF job ticket to determine if a specific value is valid.

Batch

A way to group jobs so that they can be processed together or by a device that only processes specific batches. This function is only available if the device can be configured to accept jobs that have BatchID set.

Scope

Job

Syntax

<Component ID=”*Reference ID*” Class=”Quantity” Status=”Available”
ComponentType=”FinalProduct” **BatchID**=”*name of batch or group*” />

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the Component being referenced.

BatchID

String of up to 255 characters.

Billing Code

A code to bill charges incurred.

Scope

Job

Syntax

<CustomerInfo **BillingCode**=”charge code” />

Type

Resettable, single-valued

Allowed Values

BillingCode

String of up to 255 characters.

Binding

How to bind the output of the job.

Scope

Job

Syntax

<BindingIntent ID="*Reference ID*" Class="Intent" Status="Available"
BindingType=”*type of binding*” **BindingSide**=”*side to bind*” />

Open Issue: BindingIntent is an Intent resource. There is not a generic Binding resource, there are more specific binding resources. There is not currently a Binding process with a BindingParams like the other finishing processes (Stitching, Folding, HoleMaking). This is an open issue with CIP4.

Type

Resettable, single-valued

Allowed Values

BindingType

Optional if binding type cannot be specified or can be defaulted. One of the values in the following enumeration:

SystemSpecified	Bind how system defines binding.
Tape	Apply an adhesive strip along the binding side to separate front and back covers.
SoftCover	Apply glue to the binding side of a single sheet that wraps from the front around the binding side to the back. Also referred to as perfect binding. An example of this type of binding is a paperback book.

Open Issue: SystemSpecified is currently not a value for BindingType. Propose that it be added.

BindingSide

Optional if binding side cannot be specified or can be defaulted. One of the values in the following enumeration:

Bottom	Edge that is along the x-axis of the sheet.
Left	Edge that is along the y-axis of the sheet.
Right	Edge that is opposite of the Left edge.
Top	Edge that is opposite of the Bottom edge.

Black Overprint

Automated selection of overprint for black text and/or graphics.

Scope

Job

Syntax

<RenderingParams ID=”*Reference ID*” Class=”Parameter” Status=”Available” >
<AutomatedOverprintParams **OverPrintBlackText**=”*whether to overprint black text*”
OverPrintBlackLineArt=”*whether to overprint black graphics” />*
</RenderingParams>

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the ScreeningParams being referenced.

OverPrintBlackText

Optional. Boolean. One of the following values:

true
false Default.

OverPrintBlackLineArt

Optional. Boolean. One of the following values:

true
false Default.

Class

Output class that this job is assigned to that is used to route the job to a specific device.

Scope

Job

Syntax

<Component ID="*Reference ID*" Class="Quantity" Status="Available"
ComponentType="FinalProduct" **osdp:Class**="*output class*" />

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the Component being referenced.

osdp:Class

1 alphanumeric character.

Collate

Whether the sheets in the documents and the documents in the job should be collated when multiple copies of the job are requested.

Scope

Job

Syntax

<DigitalPrintingParams ID="*Reference ID*" Class="Parameter" Status="Available"
Collate="*whether to collate or not*" />

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams being referenced.

Collate

None	Do not collate
Sheet	Collate the sheets in the documents in the job
SheetSetAndJob	Collate the sheets in the documents in the job and the documents in the job.

Example: A job contains 2 documents, A and B, each have 2 sheets, A1, A2, and B1, B2. The number of job copies requested is 3.

If Collate=None, the sheet order will be: A1A1A1 A2A2A2 B1B1B1 B2B2B2
If Collate=Sheet the sheet order will be: A1A2 A1A2 A1A2 B1B2 B1B2 B1B2
If Collate=SheetSetAndJob, the sheet order will be: A1A2B1B2 A1A2B1B2 A1A2B1B2

Comment/Description of Job

A comment that describes the job and will be carried along with the job. This information is about the job not the job ticket. Use the Job Ticket Comment or Description to include a description of the job ticket.

Scope

Job

Syntax

```
<RunList ID="Reference ID" Class="Parameter" Status="Available”>
        <Comment Name=”Description"> job comment text </Comment>
</RunList>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the RunList being referenced.

job comment text

String of up to 1024 characters.

Covers

There are two ways to include covers with a job. One, the content that is to be imaged on the covers is not included in the job, the covers are either blank or preprinted. Two, the content that is to be imaged on the covers is included in the job. For the first case, see Insert Sheet on 40 page. For the second case, the following description explains how different media can be specified for the front cover, back cover, and the document body.

Print front and/or back covers with the job, where the first one or two pages and/or the last one of two pages of the job contains the content of the front and/or back covers.

Scope

Job

Syntax

```
<Media ID=”Front cover media reference ID” Class=”Consumable” Status=”Available” />
<Media ID=”Back cover media reference ID” Class=”Consumable” Status=”Available” />
<Media ID=”Document body media reference ID” Class=”Consumable”
        Status=”Available” />
<LayoutPreparationParams ID="LayoutPreparationParams reference ID" Class="Parameter"
        Status="Available" PartIDKeys="Run RunIndex" >
        <LayoutPreparationsParams Run="Run value for document or part of document"
                <!-- Sides and Media for front cover -->
                <LayoutPreparationParams RunIndex=”0” | “0 1”
                        Sides="sides to print front cover on" >
                        <MediaRef rRef=”Front cover media reference ID” />
                </LayoutPreparationParams>
        </LayoutPreparationParams>
        <!-- Sides and Media for document body -->
        <LayoutPreparationParams RunIndex=”1~-2” | “1~-3” | “2~-2” | “2~-3”
                Sides=”sides to print document body on” >
                <Media rRef=”Document body media reference ID” />
        </LayoutPreparationParams>
```

```
    <!-- Sides and Media for back cover -->
    <LayoutPreparationParams RunIndex="-1" | “-2 -1”
        Sides="sides to print back cover on” >
        <MediaRef rRef=”Back cover media reference ID” />
    </LayoutPreparationParams>
</LayoutPreparationParams>

<!-- In the case where the media for the front and back covers is the same as the media for the
    body of the document and the entire document (covers and body) is to be printed two sided,
    the body of the document could end on the front side of a sheet, which would mean that the
    front side of the back cover would print on the back side of the last sheet of the body of the
    document. The following will force the back side of the last sheet of the body of the document
    to be blank if the body of the document ends on the front side of the sheet. It conditionally
    inserts a fill sheet. If the body of the document ends on the back side of the sheet, then a fill
    sheet is not inserted because the front side of the back cover will already be printed on the
    next sheet. -->
<LayoutElement ID="LayoutElement reference ID" Class="Parameter" Status="Available" >
    <FileSpec URL="document file name" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run">
    <!-- The following will insert a sheet before the third to the last page if necessary. -->
    <RunList Run="document's page range Run value" Pages=”0~-3”>
        <LayoutElementRef rRef="LayoutElement reference ID" />
        <InsertSheet SheetType=”FillSheet” SheetUsage=”FillForceBack”
            SheetFormat="Blank” IsWaste=”false” />
    </RunList>
    <!-- Include the last two pages which will be the back cover. -->
    <RunList Run="document's page range Run value" Pages=”-2 -1”>
        <LayoutElementRef rRef="LayoutElement reference ID" />
    </RunList>
</RunList>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the Media, LayoutPreparationParams, and RunList being referenced.

Run String that is a reference to the document’s Run value or a subset of pages in a document.

RunIndex

0 First page. Use for first RunIndex for one-sided front cover.

-1 Last page. Use for third RunIndex for one-sided back cover.

0 1 First two pages. Use for first RunIndex for two-sided front cover.

-2 -1 Last two pages. Use for third RunIndex for two-sided back cover.

1~-2 Second through second to the last pages. Use for second RunIndex for document body when front and back covers are one-sided.

1~-3 Second through third to the last pages. Use for second RunIndex for document body when front cover is one-sided and back cover is two-sided.

2~-2 Third through second to the last pages. Use for second RunIndex for document body when front cover is two-sided and back cover is one-sided.

2~-3 Third through third to the last pages. Use for second RunIndex for document body when front and back covers are two-sided.

Sides

One of the values in the following enumeration:

OneSidedBackFlipX	Image on back side of the media where the page content is placed such that binding can occur along the x-axis. This is equivalent to tumble.
OneSidedBackFlipY	Image on back side of the media where the page content is placed such that binding can occur along the y-axis.
OneSidedFront	Image on front side of the media.
TwoSidedFlipX	Image on front and back sides of media where the page content is placed such that binding can occur along the x-axis. This is equivalent to tumble.
TwoSidedFlipY	Image on front and back sides of media where the page content is placed such that binding can occur along the y-axis.

URL

Location of the document. See Document File Name on page 31 for usage notes.

Pages

Always “0-~3” for the first occurrence of Pages and and “-2 -1” for the second.

SheetType

Always set to FillSheet.

SheetUsage

Always set to FillForceBack.

SheetFormat

Always set to Blank.

IsWaste

Always set to false.

Note: Valid combinations of Pages and Sides are:

- 0 and OneSidedFront to print the first page on the front side of the front cover.
- 0 and OneSidedBackFlipX or OneSidedBackFlipY to print the first page on the back side of the front cover.
- -1 and OneSidedBackFlipX or OneSidedBackFlipY to print the last page on the back side of the back cover.
- -1 and OneSidedFront to print last page on the front side of the back cover.
- 0 1 and TwoSidedFlipX or TwoSidedFlipY to print the first two pages on the front and back sides of the front cover.
- -2 -1 and TwoSidedFlipX or TwoSidedFlipY to print the last two pages on the front and back sides of the back cover.
- 1~-2, 1~-3, 2~-2, 2~-3, and any value of Sides to print the body of the document as specified by Sides.

Cutting

How to cut the output of the document.

Scope

Document

Syntax

To perform a separation cut:

```
<CuttingParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run" >
        <CuttingParams Run="document's Run value" >
                <Cut
                        StartPosition="0 0"
                        WorkingPath="x offset  or y offset to cut"
                        WorkingDirection="Top"
                />
            </CuttingParams>
        </CuttingParams>
```

To perform a perforation cut:

```
        <PerforatingParams ID="Reference ID" Class="Parameter" Status="Available"
            PartIDKeys="Run" >
            <PerforatingParams Run="document's Run value" >
                <Perforate>
                    StartPosition="0 0"
                    WorkingPath="x offset or y offset to cut"
                    WorkingDirection="Top"
                />
            </PerforatingParams>
        </PerforatingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the PerforatingParams being referenced.

Run

String that is the reference to the document’s Run value.

StartPosition

Always set to "0 0" because starting position cannot be specified otherwise.

Open Issue: Propose that StartPosition be optional and if not specified its value is either system-defined or "0 0".

WorkingPath

Offset from reference edge, in millimeters, in which to cut. Either an x offset or a y offset can be specified (the other offset must be set to 0), but not both since the cut is performed parallel to an edge.

WorkingDirection

Direction from which cutting is performed. Always set to “Top”.

Open Issue: WorkingDirection is required by JDF so it must exist, but it is not needed if the default is system-defined. Propose that WorkingDirection be optional.

Delivery Method

Specifies what the printer is to do with the job once it receives it. This affects how the printer device is to manage the job and not how the job is managed when it is in a spooled queue.

Scope

Job

Syntax

```
    <DigitalPrintingParams ID="Reference ID"  Class="Parameter" Status="Available
        osdp:DeliveryMethod="what printer is to do with job" />
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams being referenced.

osdp:DeliveryMethod

One of the values in the following enumeration:

PrintNow	Print the job immediately and then delete it.
PrintCopyAndHold	Print one copy of the job, hold it, and then delete it after it is printed.
HoldUntilPIN	Hold the job until the user enters PIN and then delete it. See PIN on page 55 for a description of how to specify a PIN.
PrintLater	Hold the job so that it can be printed later and then delete it after it is printed.
PrintNowAndSave	Print the job now and save it so it can be selected for re-printing later. The job must be manually deleted.
Save	Do not print the job, but save it so it can be selected for printing later. The job must be manually deleted.

Destination or Physical Printer Requested

Printer or destination device ID which is the name of the physical device that will process the job, document, or range of pages. The model number can also be specified.

Scope

Page

Syntax

```
<Device ID="Reference ID"  Class="Implementation" Status="Available
        PartIDKeys="Run RunIndex" >
        <Device Run="document's Run value" >
                <Device RunIndex="pages and/or range of pages to process on the device"
                        DeviceID="name or ID of physical device"
                        Manufacturer="manufacturer of physical device"
                        ModelNumber="model number of physical device" />
        </Device>
</Device>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the Device being referenced.

Run

String that is the reference to the document's Run value.

RunIndex

x	Single pages that the device is to process. Zero or more page numbers can be specified.

x~y	Range of pages that the device is to process. Zero of more page ranges can be specified.

DeviceID

String of up to 255 characters. Some systems only allow up to 8 characters.

Manufacturer

Optional. String of up to 255 characters.

ModelNumber

Optional. String of up to 255 characters.

Document File Format

Identifies the format of the data in the document in a job.

Scope

Document

Syntax

```
<LayoutElement ID="LayoutElement reference ID" Class="Parameter" Status="Available" >
        <FileSpec MimeType="data format" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run" >
        <RunList Run="document's Run value" >
                <LayoutElementRef rRef="LayoutElement reference ID" />
        </RunList>
</RunList>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the LayoutElement and RunList being referenced.

Run

String that is the reference to the document’s Run value.

MimeType

String. Some typical values:

ascii
dbcs-ascii
ditroff
d630
epson
gof
hpgl *or* hp-gl
iso-6429
image/jpeg
kgl
lcds
afp-line-data
metacode
modca-p *or* afpds
other
passthru
pcl *or* hppcl *or* hp-pcl
pcl6
application/pdf
application/postscript
ppds
ppml
sap
sap-abap
text/plain
image/tiff

Document File Name

Name of the document (file) in a job.

Scope

Document

Syntax

For jobs that contain a single document (file):

```
<LayoutElement ID="LayoutElement reference ID" Class="Parameter"
    Status="Available" >
    <FileSpec URL="document file name" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run">
    <RunList Run="document's Run value">
        <LayoutElementRef rRef="LayoutElement reference ID" />
    </RunList>
</RunList>
```

For jobs that contain multiple documents (files):

```
<LayoutElement ID="LayoutElement 1 reference ID" Class="Parameter"
    Status="Available" >
    <FileSpec URL="document 1 file name" />
</LayoutElement>
<LayoutElement ID="LayoutElement 2 reference ID" Class="Parameter"
    Status="Available" >
    <FileSpec URL="document 2 file name" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run" >
    <RunList Run="document's Run value" >
        <LayoutElementRef="LayoutElement 1 reference ID" />
    </RunList>
    <RunList Run="document's Run value" >
        <LayoutElementRef="LayoutElement 2 reference ID" />
    </RunList>
</RunList>
```

Type

Initially settable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the LayoutElement and RunList being referenced.

Run

String that is the reference to the document's Run value.

URL

Optional. Location of the document.

If FileSpec is specified without URL then the JDF job ticket will be used for all of the document(s) that are grouped and sent along with the job ticket. The RunListLink must specify a PipeProtocol, as follows:

```
…
<ResourceLinkPool>
    RunListLink rRef=”RunList reference ID” Usage=”Input”
        CombinedProcessIndex=”0 1”
        PipeProtocol=”AccompanyingFiles” />
    …
</ResourceLinkPool>
<ResourcePool>
```

```
                <LayoutElement ID="LayoutElement reference ID" Class="Parameter"
                        Status="Available" >
                        <FileSpec />
                </LayoutElement>
                …
        </ResourcePool>
        …
```

File Type

Identifies the type of file that the document is. If the file type is not specified for a document then by default the document is assumed to be a printable document. If the document is not a printable document then, any JDF function that is specified for that document is ignored.

Scope

Document

Syntax

```
<LayoutElement ID="LayoutElement reference ID" Class="Parameter"
        Status="Available" >
                <FileSpec osdp:Type="file type" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run">
        <RunList Run="document's Run value">
                <LayoutElementRef rRef="LayoutElement reference ID" />
        </RunList>
</RunList>
```

Type

Initially settable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the LayoutElement and RunList being referenced.

Run

String that is the reference to the document's Run value.

osdp:Type

One of the values in the following enumeration:

F~~f~~ileR~~-r~~eference
F~~f~~ont
P~~p~~rintable

Note: All of the above types are for non-printable documents except for the printable type.

Fit Policy

What to do when the page content does not fit on the media.

Scope

Page

Syntax

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run RunIndex" >
        <LayoutPreparationParams Run="document's Run value" >
                <LayoutPreparationParams
                        RunIndex="pages and/or range of pages to apply fit policy to" >
                <FitPolicy SizePolicy="what to do when content doesn't fit" />
                <ImageShift
                    PositionX="how to horizontally position the content"
                    PositionY="how to vertically position the content" />
            </LayoutPreparationParams>
        </LayoutPreparationParam>
    </LayoutPreparationParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document's Run value.

RunIndex

x	Single pages to apply fit policy to. Zero or more page numbers can be specified.
x~y	Range of pages to apply fit policy to. Zero of more page ranges can be specified.

SizePolicy

One of the values in the following enumeration:

FitToPage	Scale the contents up or down to fit on the media.
ClipToMaxPage	Clip the contents to fit on the media.

PositionX

One of the values in the following emuneration:

Center	Center the contents horizontally without regard to the limitations of the media's printable area.
Left	Position the left edge of the contents to be coincident with the left edge of the media's printable area.
Right	Position the right edge of the contents to be coincident with the right edge of the media's printable area.
None	Place the contents where the print data specifies. This is the default.

PositionY

One of the values in the following emuneration:

Bottom	Position the bottom edge of the contents to be coincident with the bottom edge of the media's printable area.
Center	Center the contents vertically without regard to the limitations of the media's printable area.
Top	Position the top edge of the contents to be coincident with the top edge of the media's printable area.
None	Place the contents where the print data specifies. This is the default.

Folding

How to fold the output.

Scope

Page

Syntax

To perform a z-fold operation:

```
<FoldingParams ID="Reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run RunIndex" >
    <FoldingParams Run="document's Run value" >
```

```
        <FoldingParams RunIndex=”page or range of pages to fold”
            NoOp=”whether or not to fold”
            DescriptionType="FoldCatalog"
            FoldCatalog="F8-2"
        />
    </FoldingParams>
</FoldingParams>
```

To perform a saddle-fold operation:

```
<FoldingParams ID="Reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run RunIndex" >
    <FoldingParams Run="document's Run value" >
        <FoldingParams RunIndex=”page or range of pages to fold”
            NoOp=”whether or not to fold”
            DescriptionType="FoldCatalog"
            FoldCatalog="F4-1"
        />
    </FoldingParams>
</FoldingParams>
```

To perform a fold-in operation:

```
<FoldingParams ID="Reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run RunIndex" >
    <FoldingParams Run="document's Run value" >
        <FoldingParams RunIndex=”page or range of pages to fold”
            NoOp=”whether or not to fold”
            DescriptionType="FoldProc" >
            <Fold
                From="edge to fold"
                To="direction to fold"
                Travel="amount to offset fold"
            />
        </FoldingParams>
    </FoldingParams>
</FoldingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the FoldingParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

Either one, but not both, of the following can be specified:

x A single page to fold.

x~y A range of pages to fold together.

NoOp

Optional.

true To turn folding off. All other FoldingParams attributes are ignored.

false To use other FoldingParams attributes to specify folding. This is the default.

DescriptionType

One of the values in the following enumeration:

FoldCatalog perform a fold operation as specified by the value of FoldCatalog

FoldProc	perform a fold-in operation

FoldCatalog

Type of fold operation to perform. Only valid when DescriptionType=FoldCatalog. One of the values in the following enumeration:

F8-2	perform a z-fold operation
F4-1	perfrom a saddle-fold operation

From

Edge from which to perform folding operation. Only valid when DescriptionType=FoldProc. One of the values in the following enumeration:

Front	fold from front edge
Left	fold from left edge

To

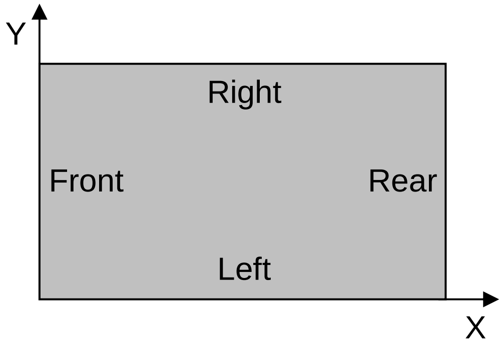
Direction in which to perform folding operation. Only valid when DescriptionType=FoldProc. One of the values in the following enumeration:

Down	fold downward
Up	fold upward

Travel

Amount to offset, in millimeters, from reference edge when performing folding operation. Only valid when DescriptionType=FoldProc.

double



Hold Job

Hold the job on the spooled queue.

Scope

Job

Syntax

<JDF ID=… Type=”Combined” Types=… Status=”Ready” **Activation**=**”***held status of job***”** …>

Type

Resettable, single-valued

Allowed Values

Activation

One of the values in the following enumeration:

Held	Job is held and cannot be processed
Active	Job is released and can be processed

Image Shift Back Side

Offset the logical page origin as specified, in micrometers, to the right or left and/or above or below of the physical page on the back of the sheet.

Scope

Page

Syntax

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available”
        PartIDKeys="Run RunIndex" >
        <LayoutPreparationParams Run="document's Run value" >
                <LayoutPreparationParams
                        RunIndex=”pages or ranges of pages to image shift” >
                        <ImageShift ShiftBack="x and y offset amount” />
                </LayoutPrepartionParams>
        </LayoutPreparationParams>
</LayoutPreparationParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x	Single pages to apply image shift on back side. Zero or more page numbers can be specified.
x~y	Range of pages to apply image shift to back side. Zero of more page ranges can be specified.

ShiftBack

Two ~~integers~~doubles, that are the amount to shift the image in the x and y directions.

Image Shift Front Side

Offset the logical page origin as specified, in micrometers, to the right or left and/or above or below of the physical page on the front of the sheet.

Scope

Page

Syntax

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available”
        PartIDKeys="Run RunIndex" >
        <LayoutPreparationParams Run="document's Run value" >
                <LayoutPreparationParams
                        RunIndex=”pages or ranges of pages to image shift” >
                        <ImageShift ShiftFront="x and y offset amount” />
                </LayoutPrepartionParams>
        </LayoutPreparationParams>
</LayoutPreparationParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single pages to apply image shift on front side. Zero or more page numbers can be specified.

x~y Range of pages to apply image shift to front side. Zero of more page ranges can be specified.

ShiftFront

Two ~~integers~~doubles, that are the amount to shift the image in the x and y directions.

Input Tray Name

Name of the input tray that contains the media to be used. Specifying input tray name overrides any input tray name used by the form definition.

Scope

Page

Syntax

```
<Media ID="Media reference ID" Class=”Consumable” Status=”Available”
        <Location LocationName="name of input tray" />
</Media>
<LayoutPreparationParams ID="LayoutPreparationParams reference ID" Class="Parameter"
        Status="Available PartIDKeys="Run RunIndex" >
        <LayoutPreparationgParams Run="document's Run value" >
                <LayoutPreparationParams RunIndex=”pages or ranges of pages”
                        <MediaRef rRef="Media reference ID" />
                </LayoutPreparationParams>
        </LayoutPrepartionParams>
</LayoutPreparationParams>
<DigitalPrintingParams ID="DigitalPrintingParams reference ID" Class="Parameter"
        Status="Available PartIDKeys="Run RunIndex" >
        <DigitalPrintingParams Run="document's Run value" >
                <DigitalPrintingParams RunIndex=”pages or range of pages”
                        <MediaRef rRef="Media reference ID" />
                </DigitalPrintingParams>
        </DigitalPrintingParams>
</DigitalPrintingParams>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the Media, LayoutPreparationParams, and DigitalPrintingParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single pages to get media from input tray name. Zero or more page numbers can be specified.

x~y Range of pages to to get media from input tray name. Zero of more page ranges can be specified.

LocationName

String of up to 255 characters. Some typical values are as follows:

Alternate
AutoEnvelopeFeed

AutomaticTtray
AutoSelect
AuxiliaryEnvelope
AuxiliaryPaper
Bottom
Capacity2000Sheet
ContinuousFormFeed
CurrentSelectedTray
Envelope
HighCapacityFeeder
LargeCapacity
Main
Manual
ManualEnvelopeFeed
Middle
Top
Tray-*n*   where *n* is an integer
Side

Input Tray Number

Number of the input tray that contains the media to be used to print the job. Specifying input tray number overrides any input tray number used by the form definition.

Scope

Page

Syntax

```
<Media ID="Media reference ID" Class=”Consumable” Status=”Available” >
    <Location LocID="number of input tray" />
</Media>
<LayoutPreparationParams ID="LayoutPreparationParams reference ID" Class="Parameter"
    Status="Available" PartIDKeys=”Run RunIndex” >
    <LayoutPreparationgParams Run="document's Run value" >
        <LayoutPreparationParams RunIndex=”pages or ranges of pages”
            <MediaRef rRef="Media reference ID" />
        </LayoutPrepartionParams>
    </LayoutPrepartionParams>
</LayoutPreparationParams>
<DigitalPrintingParams ID="DigitalPrintingParams reference ID" Class="Parameter"
    Status="Available" PartIDKeys=”Run RunIndex”>
    <DigitalPrintingParams Run="document's Run value" >
        <DigitalPrintingParams RunIndex=”pages or range of pages”
            <MediaRef rRef="Media reference ID" />
        </DigitalPrintingParams>
    </DigitalPrintingParams>
</DigitalPrintingParams>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the Media, LayoutPreparationParams, DigitalPrintingParms being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single pages to get media from input tray number. Zero or more page numbers can be specified.

x~y Range of pages to to get media from input tray number. Zero of more page ranges can be specified.

LocID

String that is a number in the range of 1 through 255.

Insert Folded Sheet

Insert a folded preprinted sheet in a document.

Scope

Page

Syntax

The same as the syntax for **Error! Reference source not found.** on page 40, except the Name attribute is included in the InsertSheet definition as follows:

<**InsertSheet** ~~**Name**=”*name of insert sheet* “~~ **SheetType=”InsertSheet”**

SheetUsage=”*insert sheet before or after*" IsWaste=”false” >

<Sheet **Name**=”*name of insert sheet* “ />

</InsertSheet>

and the same as the syntax for Folding on page 34 except the SheetName attribute is included in FoldingParams as follows:

<FoldingParams … PartIDKeys=”SheetName” >

<FoldingParams **SheetName**="*name of insert sheet* " … />

</FoldingParams>

Type

Resettable, multi-valued

Allowed Values

PartIDKeys

SheetName To refer to the specific sheet, by name, that is to be folded.

Name

String that is the name of the preprinted sheet that is to be folded.

Insert Sheet

Insert a blank (or preprinted) sheet before and/or after a specified page range. If any type of finishing (hole making, stitching, etc.) is specified for the page range, then the insert sheet will be included with the specified page range.

Scope

~~Page~~Document

Syntax

Following demonstrates how to place an insert sheet before or after a specific page range of a document:

<LayoutElement ID="*LayoutElement reference ID*" Class="Parameter" Status="Available” >

<FileSpec Url="*file name*" />

</LayoutElement>

<Media ID=”*Media 2 reference ID*” Class=”Consumable” Status=”Available” />

```
<RunList ID="RunList reference ID" Class="Parameter" Status="Available"
    PartIDKeys="Run" >
    ~~<LayoutElementRef rRef="LayoutElement reference ID" />~~
    <RunList Run="insert sheet Run value"
        Pages="x~y page range before or after insert sheet" >
        <LayoutElementRef rRef="LayoutElement reference ID" />
        <InsertSheet SheetType="InsertSheet" SheetUsage="where to insert sheet"
            IsWaste="false" >
            <RunList PageCopies= "number of copies of insert sheet" />
            <Sheet>
                <MediaRef rRef="Media 2 reference ID" />
            </Sheet>
        </InsertSheet>
    </RunList>
    <RunList Run="Run value for remainder of document"
        Pages="z~-1 page range after insert sheet" >
        <LayoutElementRef rRef="LayoutElement reference ID" />
     </RunList>
</RunList>
```

Following demonstrates how to place an insert sheet before or after a specific page range of a job:

… example goes here once we figure out how to specify this …

Type

Resettable, multi-valued

Allowed Values

ID and rRef

Strings of up to 255 characters that uniquely identify the LayoutElement, Media, and RunList being referenced.

Run

Strings that are the references to the Run values for the page range before/afer the insert sheet~~s~~ or the page range after the insert sheet. The value can be anything that uniquely identifies this RunList partition.

Pages

x~y Insert before (SheetUsage=Header) or after (SheetUsage=Trailer) the range of pages starting with *x* and ending with *y*. x is 0 to indicate the first page of the document. *y* is -1 to indicate the last page of the document.

z~-1 Page range after insert sheet to the end of the document.

PageCopies

Optional. Integer in the range 1 through 999. Default is 1.

SheetType

Always set to InsertSheet

SheetUsage

One of the values in the following enumeration:

Header	Insert sheet before the specified page(s)
Trailer	Insert sheet after the specified page(s)

IsWaste

Always set to false.

Interleave Sheets

Whether to interleave a printed or non-printed sheet between each sheet of a document.

Scope

Document

Syntax

```
<Media ID="Interleave media reference ID" Class="Consumable" Status="Available" />
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run" >
        <RunList Run="document's Run value" >
                <InsertSheet SheetType="SeparatorSheet"
                        SheetUsage="Interleaved"
                        SheetFormat="type of sheet to interleave" >
                        <MediaRef rRef="Interleave media reference ID" />
                </InsertSheet>
        </RunList>
</RunList>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the Media and RunList being referenced.

Run

String that is the reference to the document's Run value.

SheetType

Always set to Separator.

SheetUsage

Always set to Interleaved.

SheetFormat

One of the values in the following enumeration:

Duplicate	Print same content as previous page onto each interleave sheet.
Blank	Do not print on the interleave sheet. Sheet may be blank or pre-printed.

Job Created By

Name or user ID of the person who submitted the job.

Scope

Job

Syntax

```
<AuditPool>
        <Created Author="name or user ID of person who created job" />
</AuditPool>
```

Type

Initially settable, single-valued

Allowed Values

Author

String of up to 255 characters. This may contain the user ID and hostname in the form *userID@hostname.*

Job Creation Date/Time

Date and time that the job was submitted.

Scope

Job

Syntax

<AuditPool>
<**Created TimeStamp**=”*date and time*” />
</AuditPool>

Type

Initially settable, single-valued

Allowed Values

TimeStamp

dateTime of the form *YYYY-MM-DD*T*hh*:*mm*:*ss+/-ohh*:*omm*. See Appendix A - Complex Data Types on page 68 for a description of the dateTime data type.

Job ID

Text or number that uniquely identifies the job.

Scope

Job

Syntax

<JDF ID=… Type=”Combined” Types=… Status=”Ready” **JobID**=**”***job indentifer***”** …>

Type

Resettable, single-valued

Allowed Values

JobID

String of up to 255 characters.

Job Modification Date/Time

Date and time when the job was modified.

Scope

Job

Syntax

<AuditPool>
<**Modified TimeStamp**=”*date and time*” />
</AuditPool>

Type

Initially settable, multi-valued

Allowed Values

TimeStamp

dateTime of the form *YYYY-MM-DD*T*hh*:*mm*:*ss+/-ohh*:*omm*. See Appendix A - Complex Data Types on page 68 for a description of the dateTime data type.

Job Modified By

Name or user ID of the person who modified the job. The hostname may also be specified.

Scope

Job

Syntax

<AuditPool>
<**Modified Author**=”name or user ID of person *who modified job”*
osdp:HostName=”*hostname of user ID” />*
</AuditPool>

Type

Initially settable, multi-valued

Allowed Values

Author

String of up to 255 characters. This may contain the user ID and hostname in the form *userID@hostname.*

osdp:HostName

Optional. String of up to 255 characters. If not present then Author may contain the combined userID and hostname in the form *userID@hostname*.

Job Name

User friendly name of the job.

Scope

Job

Syntax

<CustomerInfo **CustomerJobName**="*job name*" />

Type

Resettable, single-valued

Allowed Values

CustomerJobName

User friendly name of the job of up to 255 characters.

Job Priority

Scheduling priority of job.

Scope

Job

Syntax

<NodeInfo **JobPriority**="*priority*" />

Type

Resettable, single-valued

Allowed Values

JobPriority

Integer in the range of 1 through 100, where 1 is the lowest priority and 100 is the highest priority.

Job Ticket Comment or Description

Text that describes what the job ticket will be used for or what function it provides.

Scope

n/a

Syntax

<**Comment Name=”Description”**> *job ticket comment text* </Comment>

Type

Resettable, single-valued

Allowed Values

Name

Always set to Description.

job ticket comment text

String of up to 1024 characters.

Job Ticket ID

Unique job ticket identifier.

Scope

n/a

Syntax

<**Comment Name=”JTID”**> *job ticket ID* </Comment>

Type

Resettable, single-valued

Allowed Values

Name

Always set to JTID.

job ticket ID

String of up to 255 characters.

Job Ticket Version

The version of the job ticket.

Scope

n/a

Syntax

<**Comment Name=”osdp:Version”**> *job ticket version* </Comment>

Type

Resettable, single-valued

Allowed Values

Name

Always set to osdp:Version.

job ticket version

String of the form a.b, where a is the major version number and b is the minor version number.

Jog Offset

Jog offset between copies of a job or between jobs.

Scope

Job

Syntax

To jog offset between copies of a job:

```
<DigitalPrintingParams ID="Reference ID" Class="Parameter" Status="Available"
        <Disjointing OffsetDirection="whether to job offset between job copies" />
</DigitalPrintingParams>
```

To jog offset between jobs:

```
<Component ID="Reference ID" Class="Quantity" Status="Available"
        ComponentType=”FinalProduct” >
        <Disjointing OffsetDirection="whether to jog offset between jobs " />
</Component>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams or Component being referenced.

OffsetDirection

Alternate	Perform jog offset.
None	Do not perform jog offset.

Example: There are three jobs, Job1, Job2, Job3. Three copies of Job1 are to be printed, two copies of Job2 are to be printed, and one copy of Job 3 is to be printed.

If jog offset between job copies is specified, then the offsets will occur as follows:

```
Job1 Copy1
                    Job1 Copy2
Job1 Copy3
Job2 Copy1
                    Job2 Copy2
Job3 Copy1
```

If jog offset between jobs is specified, then the offsets will occur as follows:

```
Job1 Copy1
Job1 Copy2
Job1 Copy3
                    Job2 Copy1
                    Job2 Copy2
Job3 Copy1
```

If jog offset between jobs and job copies is specified, then the offsets will occur as follows:

```
Job1 Copy1
                    Job1 Copy2
Job1 Copy3
                    Job2 Copy1
Job2 Copy2
                    Job3 Copy1
```

Media

Name and properties of the media on which to print the document.

Scope

Page

Syntax

To print the entire document using the same Media:

```
<Media ID="Media reference ID" Class=”Consumable” Status=”Available”
        BackCoating=”coating already applied to back side of media”
        DescriptiveName=”name of the media”
        Dimension=”media x and y dimensions”
        FrontCoating=”coating already applied to front side of media”
        HoleType=”number of pre-punched holes”
        ImagableSide=”side of media that may be marked”
        MediaColorName=”media color”
        MediaTypeDetails=”media type details”
        MediaSetCount=”number of media in each set”
        Opacity=”opacity of the media”
        Preprinted=”whether media is preprinted or not”
        Recycled=”whether recycled media is requested or not”
        osdp:PercentRecycled=”percentage of recycled material in media”
        Weight=”media weight”
        <Color sRGB=”color of media in RGB colorspace” />
</Media>
<LayoutPreparationParams ID="LayoutPreparationParams reference ID"
        Class="Parameter" Status="Available PartIDKeys="Run" >
        <LayoutPreparationParams Run="document's Run value" >
                 <MediaRef rRef="Media reference ID" />
        </LayoutPreparationParams>
</LayoutPreparationParams>
<DigitalPrintingParams ID="DigitalPrintingParams reference ID" Class="Parameter"
        Status="Available PartIDKeys="Run" >
        <DigitalPrintingParams Run="document's Run value" >
                <MediaRef rRef="Media reference ID" />
        </DigitalPrintingParams>
</DigitalPrintingParams>
```

To print part of the document using one Media and the other part using another Media. Some systems may not support this function.

```
<Media ID="Media 1 reference ID" Class=”Consumable” Status=”Available”
        … same options as shown above … />
</Media>
<Media ID="Media 2 reference ID" Class=”Consumable” Status=”Available”
        … same options as shown above … />
</Media>
<LayoutPreparationParams ID="LayoutPreparationParams reference ID"
        Class="Parameter" Status="Available PartIDKeys="Run RunIndex" >
        <MediaRef rRef="Media 1 reference ID" />
        <LayoutPreparationParams Run="document's Run value" >
                <LayoutPreparationParams RunIndex=”pages or ranges of pages” >
                        <MediaRef rRef="Media 2 reference ID" />
                </LayoutPreparationParams>
        </LayoutPreparationParams>
</LayoutPreparationParams>
<DigitalPrintingParams ID="DigitalPrintingParams reference ID" Class="Parameter"
        Status="Available PartIDKeys="Run RunIndex" >
        <MediaRef rRef="Media 1 reference ID" />
        <DigitalPrintingParams Run="document's Run value"
            <DigitalPrintingParams RunIndex=”pages or ranges of pages” >
                <MediaRef rRef="Media 2 reference ID" />
            </DigitalPrintingParams>
        </DigitalPrintingParams>
    </DigitalPrintingParams>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the Media, LayoutPreparationParams, and DigitalPrintingParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single page to print on media. Zero or more page numbers can be specified.

x y Range of pages to print on media. Zero of more page ranges can be specified.

BackCoatings

Optional. One of the values in the following enumeration:

None This is the default.
Gloss
HighGloss
Matte
Satin
Semigloss

DescriptiveName

String of up to 255 characters. Some typical values are as follows:

A3
A4
A4Tab
A5
B4
B5-JIS
Executive
Folio
Ledger
Letter
LetterTab
Statement
10x14in
11x15in
8x10in
8x10.5in
8x13in
8.25x13in
170x210mm
182x210mm
Custom

Dimension

Optional. Doubles for x and y dimensions expressed in points.

FrontCoatings

Optional. One of the values in the following enumeration:

None This is the default.

Gloss
HighGloss
Matte
Satin
Semigloss

HoleType

Optional if number of holes cannot be specified or can be defaulted. One of the values in the following enumeration:

R2-generic	2 pre-punched holes
R3-generic	3 pre-punched holes
R4-generic	4 pre-punched holes

ImagableSide

Optional. One of the values in the following enumeration:

Back
Both This is the default.
Front
Neither

MediaColorName

Optional. String of up to 255 characters without spaces. Some typical values are as follows:

Black
Blue
Brown
Buff
Gold
Goldenrod
Gray
osdp:Grey
Green
Ivory
MultiColor
Mustard
NoColor
Orange
Pink
osdp:Purple
Red
Silver
Turquoise
osdp:UserColor
Violet
White
Yellow
Any of the above prefixed with “Dark” or “Light”.

MediaTypeDetails

Optional. String of up to 255 characters without spaces. Some typical values are as follows:

FullCutTabs	Media with tabs that run full length of media, where only one tab is visible.
Letterhead	Media that has a letterhead.
Paper	Same as Plain.
PreCutTabs	Media with tabs that are cut so that more than one tab is visible.
Stationery	Media that is stationary.
TabStock	Medai with tabs that are either FullCutTabs or PreCutTabs. Use with MediaSetCount for number of

	tabs.
Transparency	Media is a transparency.

MediaSetCount

Optional. Integer in the range 1 through 65535.

Opacity

Optional. One of the values in the following enumeration:

Opaque
Transparent
osdp:Translucent

osdp:PercentRecycled

Optional. Integer between 1 and 100.

Preprinted

Optional. Boolean. One of the following values:

true
false

Recycled

Optional. Boolean. One of the following values:

true
false

sRGB

Color and sRGB are optional. Value is of the form “*r g b*”, *w*here *r, g,* and *b* are values in the range from 0.0 through 1.0. The value 0.0 is no intensity and 1.0 is full intensity. Black is “0.0 0.0 0.0” and white is “1.0 1.0 1.0”.

Weight

Optional. Double that is weight of media in grams per square meter.

Notification

Specifies who should be notified when certain events pertaining to the job occur.

Scope

Job

Syntax

```
<NodeInfo>
        <NotificationFilter
                Classes=”Event”
                Types=”events to deliver”>
                osdp:ChannelType=”how to deliver notification”
                osdp:Locator=”where to deliver notification ”
                <Comment
                        Language=”language code or language and country code”
                        Name=”Description"> comment text to accompany notification
                </Comment>
        </NotificationFilter>
</NodeInfo>
```

Type

Resettable, multi-valued (some systems only support up to 4) so multiple Notification elements can be present.

Allowed Values

Classes

Always set to Event.

Types

Optional. Strings containing event identifiers as defined by the system. Each event identifier must have the “osdp:” prefix.

Note: Ignored by systems where the type of events delivered are pre-defined and cannot be specified.

osdp:ChannelType

Optional if type of delivery is system specified or can be defaulted. One of the values in the following enumeration:

Email
File
FileAddTo
JobLog
Message
SAPcb
None

osdp:Locator

Optional if osdp:ChannelType is JobLog, SAPcb, or None. One or more can be specified where the individual addresses, names, or IDs are separated by spaces. One of the following which identifies where notification is to be sent. All of these are not supported by all systems.

1) Address (*name@node*)
2) Directory and file name
3) User ID of person

The user ID of a person must be specified in one of the following formats:

- *nodename*.*userid*
- *userid*

Comment

Language

Optional. String that represents either a language in the form “en” or a language and country code in the form “en-US”. If not specified defaults to system specified.

Name

Always set to Description.

comment text to accompany notification

Optional if comment cannot be specified. String of up to 255 characters
Note: Ignored by systems that don’t allow a comment to be specified.

Number of Copies

Number of copies of the job to print.

Scope

Job

Syntax

<ComponentLink ~~ID=”*Reference ID*” Class=”Quantity” Status=”Available” ComponentType=”FinalProduct”~~ rRef="*ID of Component resource*" Usage="Output" CombinedProcessIndex=*"index of Digital Printing process"* **Amount**=”*number of copies*”/>

Type

Resettable, single-valued

Allowed Values

~~ID~~rRef

String of up to 255 that uniquely identifies the Component resource being ~~referenced~~ linked.

Amount

Optional. Integer in the range of 1 through 32767. If not specified, then the result is as if 1 is specified.

~~ComponentType~~

~~Always set to FinalProduct.~~

Number Up

Number of pages to print on a single side of the sheet of paper.

Scope

Page

Syntax

<LayoutPreparationParams ID=”*Reference ID*” Class=”Parameter” Status=”Available”
PartIDKeys=”Run RunIndex”>
<LayoutPreparatoinParams Run=”*document’s Run value” >*
<LayoutPreparatoinParams RunIndex=”*range of pages”*
NumberUp=”*number of pages along x axis and y axis*” />
</LayoutPreparationParams>
</LayoutPreparationParams>

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x~y A range of pages to apply number up to.

NumberUp

1 1 1 page down (y-axis) and 1 page across (x-axis) on each side of a sheet of paper

1 2 1 page down (y-axis) and 2 pages across (x-axis) on each side of a sheet of paper

2 2 2 pages down (y-axis) and 2 pages across (x-axis) on each side of a sheet of paper

Order Pages

How the pages in a document in a job are to be ordered.

Scope

Document

Syntax

<LayoutPreparationParams ID=”*Reference ID*” Class=”Parameter” Status=”Available
PartIDKeys=”Run” >
<LayoutPreparationParams Run=”*document’s Run value*”
PageOrder=”*how to order pages*” />
</LayoutPreparationParams>

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document's Run value.

PageOrder

One of the values in the following enumeration:

Booklet	Order pages to be in booklet order.
Reader	Do not order the pages, leave them as is.

Output Bin Name

Name of the output bin to which output will be directed. Specifying output bin name overrides any output bin name used by the form definition.

Scope

Page

Syntax

```
<DigitalPrintingParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run RunIndex" >
        <DigitalPrintingParams Run="document's Run value" >
                <DigitalPrintingParams RunIndex="pages or ranges of pages"
                        OutputBin="output bin name" />
        </DigitalPrintingParams>
 </DigitalPrintingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams being referenced.

Run

String that is the reference to the document's Run value.

RunIndex

x	Single pages to output to bin name. Zero or more page numbers can be specified.
x~y	Range of pages to output to bin name. Zero of more page ranges can be specified.

OutputBin

String of up to 255 characters without spaces. Some typical values are as follows:

Bottom	
FaceDown	
FaceUp	
LargeCapacity	
Left	
Middle	
Right	
Side	
Stacker-n	Where *n* is an integer.
SystemDefined	This is the same as AutoSelect and is the default.
Top	
Tray-*n*	Where *n* is an integer.
osdp:Upper	
osdp:Main	

Output Bin Number

Number of the output bin to which output will be directed. Specifying output bin number overrides any output bin number used by the form definition.

Scope

Page

Syntax

```
<DigitalPrintingParams ID=”Reference ID” Class=”Parameter” Status=”Available”
        PartIDKeys=”Run RunIndex” >
        <DigitalPrintingParams Run=”document’s Run value” >
                <DigitalPrintingParams RunIndex=”pages or ranges of pages”
                        osdp:OutputBinNumber=”output bin number” />
        </DigitalPrintingParams>
</DigitalPrintingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x	Single pages to output to bin number. Zero or more page numbers can be specified.
x~y	Range of pages to output to bin number. Zero of more page ranges can be specified.

osdp:OutputBinNumber

Integer in the range 1 through 65535.

Page Delivery

How pages are to be delivered to the output bin or stacker.

Scope

Job

Syntax

```
<DigitalPrintingParams ID=”Reference ID” Class=”Parameter” Status=”Available”
        <PageDelivery=”how to deliver pages” />
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the DigitalPrintingParams being referenced.

PageDelivery

One of the values in the following enumeration:

SameOrderFaceUp	Page order as is with the front sides of the media up.
SameOrderFaceDown	Page order as is with the front sides of the media down.
ReverseOrderFaceUp	Page order reversed with the front sides of the media up.
ReverseOrderFaceDown	Page order reversed with the front sides of the media

Down.

PIN

Personal identification number that the user must enter at the printer device if the Delivery Method specifies that a PIN must be entered in order to print the job.

Scope

Job

Syntax

<Component ID=”*Reference ID*” Class=”Quantity” Status=”Available”
ComponentType=”FinalProduct” **osdp:PIN**=”*personal identification number*” />

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the Component being referenced.

osdp:PIN

String of numeric characters in the range 0000 through 9999. If a value for osdp:PIN is specified, the job can be submitted so that it will be held at the printer until the user enters this PIN. See Delivery Method on page 29.

Print After Time

Date and time after which a job is scheduled for printing or transmission.

Scope

Job

Syntax

<NodeInfo **FirstStart**=*”date and time*” />

Type

Resettable, single-valued

Allowed Values

FirstStart

dateTime of the form *YYYY-MM-DD*T*hh*:*mm:ss+/-ohh*:*omm*. See Appendix A - Complex Data Types on page 68 for a description of the dateTime data type.

Process Color Model

The model to be used for rendering the colorants defined in color spaces into process colorants.

Scope

Job

Syntax

<ColorantControl ID=”*Reference ID*” Class=”Parameter” Status=”Available”
ProcessColorModel=”*process color mode*” />

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the ColorantControl being referenced.

ProcessColorModel

One of the values in the following enumeration:

DeviceCMY
DeviceCMYK
DeviceGray
DeviceN
DeviceRGB
Default Device default

Project ID

Identification of the project context that this JDF belongs to. Can be used by the application that created the JDF to identify the application or a version of the application.

Scope

n/a

Syntax

<JDF ID=… Type=”Combined” Types=… Status=”Ready”
Project ID=”*application identification***”** …>

Type

Resettable, single-valued

Allowed Values

ProjectID

String of up to 255 characters.

Note: This function will be covered by 2 new attributes called AgentName and AgentVersion which will be added to the AuditPool Created element in JDF 1.2.

Punching and Hole Making

How many holes to punch and where to punch the holes in the output.

Scope

Page

Syntax

<HoleMakingParams ID=”*Reference ID*” Class=”Parameter” Status=”Available”
PartIDKeys=”Run Run~~List~~Index” >
<HoleMakingParams Run=”*document’s Run value” >*
<HoleMakingParams RunIndex=”*pages or ranges of pages to punch”*
NoOp=”*whether or not to punch holes”*
HoleType=”*number of holes to punch*”
HoleReferenceEdge=”*which edge to punch” />*
</HoleMakingParams>

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the HoleMakingParams being referenced.

NoOp

Optional.

true To turn hole making off. All other HoleMakingParams attributes are ignored.

false To use other HoleMakingParams attributes to specify hole making. This is the default.

Run

String that is the reference to the document's Run value.

RunIndex

x Single page to punch holes into. Zero or more page numbers can be specified.

x~y Range of pages to punch holes into. Zero of more page ranges can be specified.

HoleType

Optional if number of holes cannot be specified or can be defaulted. One of the values in the following enumeration:

R2-generic
R3-generic
R4-generic

HoleReferenceEdge

Optional if reference edge cannot be specified or can be defaulted. One of the values in the following enumeration:

Bottom Edge that is along the x-axis of the sheet.

Left Edge that is along the y-axis of the sheet.

Right Edge that is opposite of the Left edge.

Top Edge that is opposite of the Bottom edge.

Range of Pages

Range of pages in a document that the job ticket will be generated for.

Scope

Document

Syntax

<LayoutElement ID="*LayoutElement reference ID*" Class="Parameter" Status="Available >

<FileSpec URL="*document file name*" />

</LayoutElement>

<RunList ID="*RunList reference ID*" Class="Parameter" Status="Available" PartIDKeys="Run" >

<RunList Run="*document's Run value*" **Pages**="*pages or ranges of pages to include ~~process~~*" >

<LayoutElementRef="*LayoutElement reference ID" />*

</RunList>

</RunList>

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the LayoutElement and RunList being referenced.

Run

String that is the reference to the document's Run value.

Pages

x Page numbers to include to generate the job ticket~~ for~~. Zero or more page numbers can be specified.

x~*y* Page ranges to include to generate the job ticket ~~for~~, where x is the first page and y is the last
page. When y is -1, page x through the last page. Zero or more page ranges can be specified.

Range of Pages to Process

Range of pages in the job or document to process. The value specified must be a subset of Range of Pages on page 57. If not specified, it defaults to Range of Pages.

Scope

Document

Syntax

```
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run" >
        <RunList Run="document's Run value"
                osdp:ProcessPageRange="pages and range of pages to process" />
</RunList>
```

Type

Resettable, single-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the RunList being referenced.

Run

String that is the reference to the document's Run value.

Pages

x Page numbers that are to be processed. Zero or more page numbers can be specified.

x~*y* Page ranges to process where x is the first page and y is the last page to be processed. When y is -1, page x through the last page is processed. Zero or more page ranges can be specified.

Note: Apparently this function can be accomplished by partitioning the RunListLink.

Retain Job

Whether a job is to be retained in the spooler or queue after it is processed and how long it should be retained.

Because the Disposition attribute belongs to the FileSpec element, it can be applied to each file in a job. The system may not support treating each file differently, so either all files must have the same Dispostion and DispositionTime specified or the first occurrences will be used for all files in the job.

Scope

Job

Syntax

To retain the job forever:

```
<LayoutElement ID="LayoutElement reference ID" Class="Parameter"
        Status="Available" >
        <FileSpec Disposition="Retain" />
</LayoutElement>
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" >
```

```
        <LayoutElementRef="LayoutElement reference ID" />
    </RunList>
```

To retain the job forever if it fails:

```
    <LayoutElement ID="LayoutElement reference ID" Class="Parameter"
        Status="Available" >
        <FileSpec osdp:FailDisposition="Retain" />
    </LayoutElement>
    <RunList ID="RunList reference ID" Class="Parameter" Status="Available" >
        <LayoutElementRef="LayoutElement reference ID" />
    </RunList>
```

To retain the job for a specific amount of time:

```
    <LayoutElement ID="LayoutElement reference ID" Class="Parameter"
        Status="Available" >
        <FileSpec Disposition="Retain"
            DispositionPeriod="amount of time to retain" />
    </LayoutElement>
    <RunList ID="RunList reference ID" Class="Parameter" Status="Available" >
        <LayoutElementRef="LayoutElement reference ID" />
    </RunList>
```

To retain the job for a specific amount of time if it fails:

```
    <LayoutElement ID="LayoutElement reference ID" Class="Parameter"
        Status="Available" >
        <FileSpec osdp:FailDisposition="Retain"
            osdp:FailDispositionPeriod="amount of time to retain" />
    </LayoutElement>
    <RunList ID="RunList reference ID" Class="Parameter" Status="Available" >
        <LayoutElementRef="LayoutElement reference ID" />
    </RunList>
```

Note: For those systems where there is a separate specification of retention for jobs that fail and jobs that are successful, Disposition refers to the jobs that are successful and FailDisposition refers to the jobs that fail. For those systems where there is a single specification for all jobs, Disposition refers to all jobs and FailDisposition is ignored.

Type

Resettable, single-valued, but both Disposition and FailDisposition can be specificed at the same time.

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the LayoutElement and RunList being referenced.

Disposition

Optional if osdp:FailDispositionis present. One of the values in the following enumeration:

Retain	Retain job after it is printed for the specified time period or forever if a time period is not specified.
Unlink	Don’t retain job, delete it after it prints.
osdp:RetainImmediately	Retain job as soon as it is submitted without printing it first. Retain for the specified time period or forever if a time period is not specified.

Open Issue: Retain is currently the default for Disposition in JDF. Should the default be SystemSpecified? Or should something besides Disposition be used for Retain Job?

osdp:FailDisposition

Optional if Disposition is present. One of the values in the following enumeration:

Retain — Retain job after it is printed for the specified time period or forever if a time period is not specified.

Unlink — Don’t retain job, delete it after it prints.

DispositionPeriod

Optional. duration of the form P*y*Y*m*M*d*DT*hh*H*mm*M. See Appendix A - Complex Data Types on page 68 for a description of duration. Only valid when Disposition is present. Open Issue: DispositionPeriod is not currently supported by JDF. Propose that it be added or make it osdp:DispositionPeriod.

osdp:FailDispositionPeriod

Optional. duration as defined above. Only valid when osdp:FailedDisposition is present.

Rotate Page

Degrees of rotation of the text or image on a page with respect to the way the form is loaded into the printer.

Scope

Page

Syntax

<LayoutPreparationParams ID=”*Reference ID*” Class=”Parameter” Status=”Available
PartIDKeys=”Run RunIndex”>
<LayoutPreparationParams Run=”*document’s Run value”* >
<LayoutPreparationParams RunIndex=”*pages or ranges of pages”*
Rotate=”*degrees of rotation*” />
</LayoutPreparationParams>
</LayoutPreparationParams>

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single page to rotate. Zero or more page numbers can be specified.

x~y Range of pages to rotate. Zero of more page ranges can be specified.

Rotate

One of the values in the following enumeration:

Rotate0	
Rotate90	Rotate 90 degrees counter-clockwise
Rotate180	
Rotate270	Rotate 90 degrees clockwise

Screening Family

Vendor specific screening family to use.

Scope

Job

Syntax

```
<ScreeningParams ID=”Reference ID” Class=”Parameter” Status=”Available” >
        <ScreenSelector ScreeningFamily=”screening family name” />
</ScreeningParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the ScreeningParams being referenced.

ScreeningFamily

String of up to 255 characters. Some typical values are as follows:

Rational Tangent

Adobe Accurate

Agfa Balanced

Soft-IS

ErrorDiffusion

Sides

Whether to print on one or two sides the sheet.

Scope

Page

Syntax

To print the entire document the same:

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run RunIndex" >
        <LayoutPreparationParams Run="document's Run value" >
                <LayoutPreparationParams RunIndex=”pages or ranges of pages”
                        Sides="sides to print on" />
        </LayoutPreparationParams>
</LayoutPreparationParams>
```

Type

Resettable, sIngle-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x	Single page to print as specified by Sides. Zero or more page numbers can be specified.
x y	Range of pages to print as specified by Sides. Zero or more page ranges can be specified.

Sides

One of the values in the following enumeration:

OneSidedBackFlipX	Image on the back side of the media where the page content is placed such that binding can occur along the x-axis. This is equivalent to tumble.
OneSidedBackFlipY	Image on the back side of the media where the page content is placed such that binding can occur along the

	y-axis.
OneSidedFront	Image on the front side of the media. This is the default if Sides is not specified.
TwoSidedFlipX	Image on the front and back sides of media where the page content is placed such that binding can occur along the x-axis. This is equivalent to tumble.
TwoSidedFlipY	Image on the front and back sides of media where the page content is placed such that binding can occur along the y-axis.

Spot Color

Separation to render spot color.

Scope

Job

Syntax

```
<ColorantControl ID="Reference ID" Class="Parameter" Status="Available" >
        <ColorantParams>
                <SeparationSpec Name="separation name" />
         </ColorantParams>
</ColorantControl>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the ColoranControl being referenced.

Name

String of up to 255 characters.

Stapling and Stitching

How specific pages, documents, or job should be stapled/stitched together.

Scope

Page

Syntax

```
<StitchingParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run RunIndex"
        <StitchingParams Run="document's Run Value" >
                <StitchingParams RunIndex="range of pages to stitch together"
                        NoOp="whether or not to stitch"
                        Angle="angle of stitch"
                        NumberOfStitches="number of stitches"
                        ReferenceEdge="which edge to stitch"
                        StitchType="position of stitches" />
        </StitchingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the StitchingParams being referenced.

NoOp

Optional.

true	To turn stitching off. All other StitchingParams attributes are ignored.
false	To use other StitchingParams attributes to specify stitching. This is the default.

Run

String that is the reference to the document's Run value.

RunIndex

x~y A range of pages to stitch together.

Angle

Optional if angle cannot be specified or can be defaulted. Integer that is the angle, in degrees, of each stitch; Increasing in the counterclockwise direction. Valid values are 0 through 359, where 0 is horizontal or parallel to the x axis. The default is system specified. If StitchType is Saddle, Angle is ignored.

NumberOfStitches

Optional if number of staples cannot be specified or can be defaulted. Integer in the range of 1 through 255.

ReferenceEdge

Optional if reference edge cannot be specified or can be defaulted. One of the values in the following enumeration:

Bottom	Edge that is along the x-axis of the sheet.
Left	Edge that is along the y-axis of the sheet.
Right	Edge that is opposite of the Left edge.
Top	Edge that is opposite of the Bottom edge.

StitchType

Optional if position of staple cannot be specified or can be defaulted. One of the values in the following enumeration:

Corner	Stitch in the corner that is at the clockwise end of the ReferenceEdge.
Saddle	Stitch the on the middle fold which is on the saddle.
Side	Stitch along the reference edge. If NumberOfStitches is 1, then the stitch is placed parallel to the ReferenceEdge and in the corner that is at the clockwise end of the ReferenceEdge.

Start First Page

Start the next page after the specified range of pages on either the front or back side of the sheet. This function can be used to ensure that chapters will start on a specific side of the sheet and is only honored when two-sided printing is specified.

Scope

Page

Syntax

```
<RunList ID="RunList reference ID" Class="Parameter" Status="Available" PartIDKeys="Run">
        <!-- The following will insert a sheet after the last page of the specified page range so that
             the next page after the page range will print on the desired side of the sheet.  -->
        <RunList Run="document's page range Run value"
                Pages="range of pages before forcing a new sheet">
                <LayoutElementRef rRef="LayoutElement reference ID" />
                <InsertSheet SheetType="FillSheet"
                SheetUsage="whether to force to back or front side" SheetFormat=Blank"
                IsWaste="false" />
        </RunList>
        … Include as many partitioned RunLists here as needed in the form as shown above.
        … There should be one for each page range after which the next page will be forced to
```

```
          … print on the specified side.
          <!-- The last page range is for the end of the document. -->
          <RunList Run="document's page range Run value"
                    Pages=”last page range of document” >
                    <LayoutElementRef rRef="LayoutElement reference ID" />
          </RunList>
</RunList>
```

Type

Resettable, multi-valued

Allowed Values

ID and rRef

Strings of up to 255 that uniquely identify the RunList being referenced.

Run

String that is a reference to a document or to a ~~the~~ subset of pages in a document.

Pages

x~y Range of pages that are to occur before forcing a new sheet. The last page range begins with the last page to force to a specific side through the end of the document (-1).

SheetType

Always set to FillSheet.

SheetUsage

One of the values in the following enumeration:

FillForceBack If necessary, add a fill page to the back of the last page of the specified page range, so that the next page will print on a new sheet.

FillForceFront If necessary, add a fill page to the front of the last page of the specified page range, so that the next page will print on a new sheet.

SheetFormat

Always set to Blank.

IsWaste

Always set to false.

Example: When printing two sided, to ensure that pages 3 and 6 are printed on the front side of a sheet, set SheerUsage to FillForceBack, and specify the following page ranges:

x	y	Result
0	2	A fill page will be inserted for the back side of page 2, if page 2 is on the front side.
3	5	A fill page will be included for the back side of page 5, if page 5 is on the front side.
6	-1	Last page range of the document

Start, Separator/Slip, End Sheets

Whether start, separator/slip, and/or end sheets are to be included with the job. If any type of finishing (hole making, stitching, etc.) is specified for any part of the job, the start, separator/slip, and end sheets will not be included in the finishing.

Scope

Job

Syntax

<**Media ID**="*Media reference ID*" Class=”Consumable” Status=”Available” … />

```
<LayoutPreparationParams ID="Reference ID" Class="Parameter" Status="Available" >
        <InsertSheet SheetType="type of insert sheet to include"
                SheetUsage="where to include insert sheet"
                        SheetFormat="what it to appear on separator sheet" ~~/~~>
                <Sheet>
                        <MediaRef rRef="Media reference ID" />
                </Sheet>
        </InsertSheet>
</LayoutPreparationParams>
```

Type

Resettable, multi-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the LayoutPreparationParams being referenced.

SheetFormat

Optional. One fo the values in the following enumeration:

Blank	Content of Separator sheet is to be blank.
SystemSpecified	Content of Separator sheet is be defined by the system. This is the default.

SheetType

One of the values in the following enumeration:

JobSheet	Include a sheet that delimits the job
SeparatorSheet	Include a sheet that delimits each copy of each document in the job.

SheetUsage

One of the values in the following enumeration:

Trailer
Include an insert sheet at the end of the job when SheetType=JobSheet or at the end of each copy of each document in the job when SheetType=SeparatorSheet.

Slip
Include an insert sheet between each copy of each document in the job~~.~~ ~~Only valid~~ when SheetType=Separator or between copies of the job when SheetType=JobSheet.
Note: Currently in the JDF 1.1 Spec SheetUsage=Slip and SheetType=JobSheet is not a valid combination. Can this be added to allow separator sheets between job copies?

Header
Include an insert sheet at the start of the job when SheetType=JobSheet or at the start of each copy of each document in the job when SheetType=SeparatorSheet.

Trimming

Perform a trimming operation on the final product.

Scope

Page

Syntax

```
<TrimmingParams ID="Reference ID" Class="Parameter" Status="Available"
        PartIDKeys="Run RunIndex"
        <TrimmingParams Run="document's Run value"
                <TrimmingParams RunIndex="pages or range of pages to trim"
                        NoOp="whether or not to trim "
                        TimmingType="SystemSpecified" />
        </TrimmingParams>
```

```
</TrimmingParams>
```

Type

Resettable, single-valued

Allowed Values

ID

String of up to 255 that uniquely identifies the TrimmingParams being referenced.

Run

String that is the reference to the document’s Run value.

RunIndex

x Single page to trim or not. Zero or more page numbers can be specified.

x y Range of pages to trim or not. Zero or more page ranges can be specified.

NoOp

Optional.

true To turn trimming off. All other TrimmingParams attributes are ignored.

false To use other TrimmingParams attributes to specify trimming. This is the default.

TrimmingType

Always set to SystemSpecified to perform trimming operation.

8. Other items of interest

JMF

JMF (Job Messaging Format) is a well-defined syntax that allows JDF processes, devices, and MIS systems to effectively and efficiently communicate. It provides a wide range of capabilities to facilitate interaction between various aspects of workflow, from simple unidirectional notification through issuing commands.

JMF is encoded in XML. JMF and JDF have an inherently different structure. JMF uses the unique name **JMF** as its root-element name.

JMF encodes messages of several types. There are message elements that contain and convey content and there are message elements that describe the way in which these element types can be used to establish communication. JMF messages can be used to perform queries, receive responses, signal status changes, issue commands, and acknowledge commands. JMF can be set up so that messages can be sent and not acknowledged by the receiver or subscriptions can be set up so that there are persistent channels and messages are sent whenever a specified condition is met.

9. Appendix A - Complex Data Types

1) **dateTime** is defined by ISO8601 and is of the form *YYYY-MM-DD*T*hh*:*mm:ss+/-ohh*:*omm* where:

YYYY	Four-digit year
MM	Two-digit month (01=January, etc.)
DD	Two-digit day of month (01 through 31)
T	Required designator specifying time
hh	Two-digit hour using 24 hour clock (00 through 23)
mm	Two-digit minutes (00 through 59)
ss	Two-digit seconds (00 through 59)
+/-	Whether local time offset is after(+) or before(-) Coordinated Universal Time (UTC)
ohh:*omm*	Hours and minutes local time is offset from Coordinated Universal Time (UTC)

For example 2001-11-09T16:40:00-07:00 is November 9th 2001 4:40 PM Mountain Standard time (MST is 7 hours behind Greenwich time)

2) **duration** is defined by ISO8601 and is of the form P*y*Y*m*M*d*DT*hh*H*mm*M, where:

	P Required delimiter
y	How many years to retain job
Y	Required year delimiter
m	How many months to retain job
M	Required month delimiter
d	How many days to retain job
D	Required day delimiter
T	Required time delimiter
hh	How many hours to retain job
H	Required hour delimiter
mm	How many minutes to retain job
M	Required minute delimiter

For example P1Y2M3DT05H30M is a duration of 1 year, 2 months, 3 days, 5 hours, and 30 minutes.

10. Appendix B - JDF Examples

1) Following is an example of how a simple JDF job ticket would look:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Simple JDF Job Ticket  -->

<!-- Identify the processes that are to be executed and the order that they will be executed. -->
<JDF ID="SimpleJT" Status="Ready" Type="Combined" Types="LayoutPreparation Imposition Interpreting
    Rendering DigitalPrinting Stitching HoleMaking Folding" xmlns="http://www.cip4.org/JDFSchema_1_1"
    xmlns:osdp="private Open Source Digital Printing XML namespace" Version="1.1" >

    <!-- Identify each resource as an input to or output from a process(es) and link each to its
         associated process(es) -->
    <ResourceLinkPool>
            <RunListLink rRef="RL" Usage="Input"
                    CombinedProcessIndex="0 1" />
            <LayoutPreparationParamsLink rRef="LPP" Usage="Input"
                    CombinedProcessIndex="0" />
            <MediaLink rRef="White" Usage="Input"
                    CombinedProcessIndex="4" />
            <DigitalPrintingParamsLink rRef="DPP" Usage="Input"
                    CombinedProcessIndex="4" />
            <!-- Print 5 copies -->
            <ComponentLink rRef="CP" Usage="Output"
                    CombinedProcessIndex="4" Amount="5" />
            <StitchingParamsLink rRef="SP" Usage="Input"
                    CombinedProcessIndex="5" />
            <HoleMakingParamsLink rRef="HMP" Usage="Input"
                    CombinedProcessIndex="6" />
            <FoldingParamsLink rRef="FP" Usage="Input"
                        CombinedProcessIndex="7" />
    </ResourceLinkPool>

    <!—The Resouce Pool contains the definition of all of the resources to be used. -->
    <ResourcePool>
            <!-- Identify file to print -->
            <LayoutElement ID="InputFile" Class="Parameter" Status="Available" >
                    <FileSpec URL="file:///d:/myPrintFile.afp" />
            </LayoutElement>

            <!-- Indentify pages to print. "0~-1" is all pages -->
            <RunList ID="RL" Class="Parameter" Status="Available" Pages="0~-1" >
                    <LayoutElementRef rRef="InputFile" />
            </RunList>

            <!-- Print duplex and get media information needed for layout -->
            <LayoutPreparationParams ID="LPP" Sides="TwoSidedFlipY" >
                    <MediaRef rRef="WhiteTray1"/>
            </LayoutPreparationParams>

            <!-- Define media -->
            <Media ID="WhiteTray1" Class="Consumable" Status="Available"
                    DescriptiveName=”Letter” Dimension="612 792"
                    Weight="150" MediaColorName="White" >
                    <Location LocationName="Tray1" />
            </Media>

            <!-- Use media from tray 1 and collate output -->
            <DigitalPrintingParams ID="DPP" Class="Parameter" Status="Available"  Collate="Sheet" >
                    <MediaRef rRef="WhiteTray1" />
            </DigitalPrintingParams>

            ~~<!-- Print 5 copies -->~~
            <Component ID="CP" Class="Quantity" Status="Available" ComponentType="FinalProduct"
                    ~~Amount="5"~~ />

            <!-- Stitch in top left corner -->
            <StitchingParams ID="SP" StitchType="Corner" ReferenceEdge="Left" NumberOfStitches="1" />

            <!-- 3 hole punch along left edge -->
            <HoleMakingParams ID="HMP" HoleType="R3-generic"  HoleReferenceEdge="Left" />

            <!-- Fold in half  (saddle fold) -->
            <FoldingParams ID="FP" DescriptionType="FoldCatalog FoldCatalog="F4-1" />
    </ResourcePool>
</JDF>
```

2) Following is an example of the JDF that specifies different attributes for each document in a job.

To specify that a different medium is to be used for each document in the job, partitioning the RunList is used to reference each document (file) defined in the RunList:

```
...
<ResourcePool>
    .....
    <!-- Define the files that are to be processed -->
    <LayoutElement ID="File1" Class="Parameter" Status="Available" >
        <FileSpec URL=" file:///d:/my1stPrintFile.ps" />
    </LayoutElement>
    <LayoutElement ID="File2" Class="Parameter" Status="Available" >
        <FileSpec URL=" file:///d:/my2ndPrintFile.ps" />
    </LayoutElement>

    <!-- Partition the RunList so that the files can be individually referenced -->
    <RunList ID="RL" Class="Parameter" Status="Available" PartIDKeys="Run" >
        <RunList Run="1" >
            <LayoutElementRef rRef="File1" />
        </RunList>
        <RunList Run="2" >
            <LayoutElementRef rRef="File2" />
        </RunList>
    </RunList>

    <-- Define the media and the trays where the media is to be pulled from -->
    <Media ID="WhiteTop" Class="Consumable" Status="Available"
      DescriptiveName=”Letter” MediaColorName="White" >
      <Location LocationName="Top" />
    </Media>
```

```
        <Media ID="BlueBottom" Class="Consumable" Status="Available"
          DescriptiveName=”Letter” MediaColorName="Blue" >
          <Location LocationName="Bottom" />
        </Media>

        <! -- Partition so that the correct media is referenced for each file -->
        <LayoutPreparationParams ID="LP" PartIDKeys="Run" >
            <LayoutPreparationParams Run=”1” >
              <MediaRef rRef="WhiteTop" />
            </LayoutPreparationParams>
            <LayoutPreparationParams Run=”2” >
              <MediaRef rRef="BlueBottom" />
            </LayoutPreparationParams>
        </LayoutPreparationParams>
        <DigitalPrintingParams ID="DPP" PartIDKeys="Run" >
            <DigitalPrintingParams Run=”1” >
              <MediaRef rRef="WhiteTop" />
            </DigitalPrintingParams>
            <DigitalPrintingParams Run=”2” >
              <MediaRef rRef="BlueBottom" />
            </DigitalPrintingParams Params>
        </ DigitalPrintingParamsParams>
        …
    </ResourcePool>
    ...
```

3) Following is an example of the JDF that is required to specify that the document in a job be printed simplex and the second document be printed duplex.

```
    ...
    <ResourcePool>
        .....
        <!-- Define the files that are to be processed -->
        <LayoutElement ID="File1" Class="Parameter" Status="Available" >
            <FileSpec URL=" file:///d:/my1stPrintFile.ps" />
        </LayoutElement>
        <LayoutElement ID="File2" Class="Parameter" Status="Available" >
            <FileSpec URL=" file:///d:/my2ndPrintFile.ps" />
        </LayoutElement>

        <!-- Partition the RunList so that the files can be individually referenced -->
        <RunList ID="RL" Class="Parameter" Status="Available" PartIDKeys="Run" >
            <RunList Run="1" >
                <LayoutElementRef rRef="File1" />
            </RunList>
            <RunList Run="2" >
                <LayoutElementRef rRef="File2" />
            </RunList>
        </RunList>

        <! -- Partition so that the documents can be printed on the correct sides of the paper  -->
        <LayoutPreparationParams ID="LP" PartIDKeys="Run" >
            <LayoutPreparationParams Run=”1” Sides=”OneSidedFront “ />
            <LayoutPreparationParams Run=”2” Sides=”TwoSidedFlipY” />
        </LayoutPreparationParams>
          …
        </ResourcePool>
        …
```

To specify that the JDF be applied to the entire job (to all documents in the job), the following changes to the above should be made:

- Remove the "PartIDKeys" attribute from the parent (root) LayoutPreparationParams resource.
- Remove the child (nested) resources.
- Add the Sides specification to the remaining LayoutPreparationParams resource.

So, to specify that all files in a job be printed simplex, the JDF would appear as follows:

```
    ...
    <ResourcePool>
        .....
        <!-- Define the files that are to be processed -->
        <LayoutElement ID="File1" Class="Parameter" Status="Available" >
            <FileSpec URL=" file:///d:/my1stPrintFile.ps" />
        </LayoutElement>
        <LayoutElement ID="File2" Class="Parameter" Status="Available" >
            <FileSpec URL=" file:///d:/my2ndPrintFile.ps" />
        </LayoutElement>

        <!-- Partition the RunList so that the files can be individually referenced -->
        <RunList ID="RL" Class="Parameter" Status="Available" PartIDKeys="Run" >
            <RunList Run="1" >
                <LayoutElementRef rRef="File1" />
            </RunList>
            <RunList Run="2" >
                <LayoutElementRef rRef="File2" />
            </RunList>
        </RunList>

        <! -- Entire job is printed simplex  -->
        <LayoutPreparationParams ID="LP" Sides=”OneSidedFront “ />
        …
     </ResourcePool>
    …
```

4) Following is an example of a job ticket that includes billing information, event notification, job creation information, and output bin specificaiton:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Another simple JDF Job Ticket  -->

<!--Only need to list the required processes -->
<JDF ID="JosephinesJT" Status="Ready" Type="Combined" Types="LayoutPreparation Imposition
Interpreting
    Rendering DigitalPrinting" xmlns="http://www.cip4.org/JDFSchema_1_1"
       xmlns:osdp="private Open Source Digital Printing XML namespace" Version="1.1" >

    <Comment Name=”Description”> Output my job to the Top bin and send notification to me </Comment>

    <!-- Keep track of billing info for this job -->
        <CustomerInfo BillingCode=”1234” />
        <!-- Send default notitification events to Josephine about job -->
    <NodeInfo>
            <NotificationFilter Classes=”Event” osdp:Locator=Josephine.User@myPC” />
    </NodeInfo>

    <!-- Identify each resource as an input to or output from a process(es) and link each to its
            associated process(es) -->
    <ResourceLinkPool>
            <RunListLink rRef="RL" Usage="Input"
                        CombinedProcessIndex="0 1" />
            <DigitalPrintingParamsLink rRef="DPP" Usage="Input"
                        CombinedProcessIndex="4" />
            <!-- Print 1 copy -->
            <ComponentLink rRef="CP" Usage="Output"
                        CombinedProcessIndex="4" Amount="1" />
    </ResourceLinkPool>

    <!-- List all resources to be used. -->
        <ResourcePool>
                <!-- Identify file to print -->
            <LayoutElement ID="InputFile" Class="Parameter" Status="Available" >
                    <FileSpec URL="file:///d:/myPrintFile.afp" />
            </LayoutElement>

                <!-- Indentify pages to print. "0~-1" is all pages -->
            <RunList ID="RL" Class="Parameter" Status="Available" Pages="0~-1" >
                    <LayoutElementRef rRef="InputFile" />
            </RunList>

            <! – Output completed job to Top bin -->
            <DigitalPrintingParams ID="DPP" Class="Parameter" Status="Available"  OutputBin="Top" />

            ~~<!-- Print 1 copy -->~~
            <Component ID="CP" Class="Quantity" Status="Available" ComponentType="FinalProduct"
                    ~~Amount="1"~~ />
    </ResourcePool>
    <AuditPool>
            <!-- Who submitted the job and when it was submitted -->
            <Created Author=”Josephine User” TimeStamp=”2002-04-17T18:40:20-7:00” />
    </AuditPool>
</JDF>
```

11. JDF Mapping Addendum

The JDF function described in this specification can be mapped to the function that is supported by other products.

Part A – JDF to PODi PPML Mapping

The following table contains all of the JDF function described in this specification that is supported by the PODi PPML Job Ticket.

Description of function supported by Open Source for Digital Printing JDF	Page
Black Overprint	24
Collate	25
Document File Name	31
Folding	34
Input Tray Name	38
Jog Offset	45
Media	46
Number of Copies	51
Number Up	52
Order Pages	52
Page Delivery	54
Process Color Model	55
Punching and Hole Making	56
Screening Family	60
Sides	61
Spot Color	62
Stapling and Stitching	62
Trimming	65

The implementation of the PODi PPML Job Ticket makes use of an abstract element called ResourceUpdate. See Resource Update on page 20 for an explanation of the use of ResourceUpdate elements. Multiple ResourceUpdate elements can be defined for a resource. Each ResourceUpdate is referenced by a TICKET_REF in the PPML source file. This allows the PPML source file to dynamically change the values of the resource's attributes as the job progresses. See the PODi "PPML Job Ticketing Functional Specification" for examples of the use of ResourceUpdate elements.

Part B - JDF to IPP Mapping

The following table contains all of the JDF function described in this specification that is supported by the Internet Printing Protocol (IPP).

This table will be added later.

Part C - JDF to FSG Open Print Mapping

The following table contains all of the JDF function described in this specification that is supported by the Free Standards Group (FSG) Open Print solution.

This table will be added later.